PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
G06F 17/30

A2

(11) International Publication Number: WO 98/18092

(43) International Publication Date: 30 April 1998 (30.04.98)

(21) International Application Number: PCT/US97/18935

(22) International Filing Date: 21 October 1997 (21.10.97)

(30) Priority Data:

60/029,425	22 October 1996 (22.10.96)	US
60/028,985	22 October 1996 (22.10.96)	US

(71) Applicant *(for all designated States except US)*: TEMPEST SOFTWARE INCORPORATED [US/US]; Texas Commerce Tower, 50th floor, 600 Travis, Houston, TX 77002 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: SALTSMAN, Michael, L. [US/US]; 7306 Wovenwood Drive, Houston, TX 77041 (US). SPENCE, Luke, A. [US/US]; Apartment 2901, 1617 Fannin, Houston, TX 77002 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

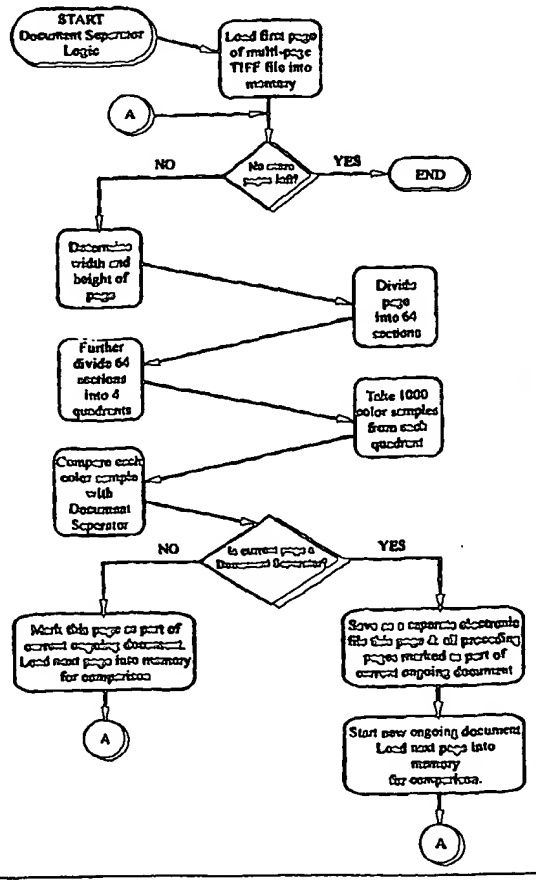
Published
Without international search report and to be republished upon receipt of that report.

09/94,72

(54) Title: METHOD AND APPARATUS FOR SCANNING AND MANAGING DOCUMENT IMAGES

(57) Abstract

A document management system wherein efficient and high-speed inputting of a voluminous number of documents is facilitated by means of a document separator, where said document separator contains a predetermined unique graphic image, which said image is interpreted by said system to perform a predetermined set of tasks. A document management system wherein said system is modular in design and permits user to select individual software components to be used with said system.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

METHOD AND APPARATUS FOR SCANNING AND MANAGING DOCUMENT IMAGES

CROSS-REFERENCE TO RELATED APPLICATIONS

The present application claims the benefit of 35 U.S.C. 111(b) provisional application Serial No. 60/029,425 filed October 22, 1996, and Serial No. 60/028,985, filed October 22, 1996, entitled Method and Apparatus for Scanning and Managing Document Images and Method and Apparatus for Computer Instruction Via Digitized Images. Both of these provisional applications are incorporated by reference, as if fully set forth herein.

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

Not applicable.

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention relates to a method and apparatus for a scanning and document management system using hardware and computer software technology. More particularly the present invention relates to the industry catering to users of document management systems and instruction of computers via digitized images.

Description of Related Art

The document management industry is constantly struggling with problems associated with efficient and cost-effective management of voluminous documents. The industry lacks cost-efficient computerized document management system capable of handling the input of a voluminous number of documents into a computer system. Thus one of the biggest stumbling blocks with the present technology is the inability to quickly and efficiently input voluminous number of documents into an imaging database. Current document management systems employ a complicated arrangement of user interfaces that require extensive training in order to adequately utilize the system. Databases with complicated user interfaces are very expensive to implement in an organization. This presents a cost barrier to an organization contemplating installation of such a system.

Another shortcoming of the state of the art is the lack of a self-configuring database. The problem to be solved is in the design and implementation of a system that allows all technical functions to be handled by the program behind the scenes and keep the user interface as simple as possible. A simple user interface allows the performance of various different tasks behind the scenes by a simple click of a mouse button. Every such step that can be

performed behind the scenes cuts down the cost of implementation and use. For example, the scanning of documents into the system involves many different tasks. The economic advantage of such an improvement over the prior art is the ability to use low-level employees to run the system without the need for any specialized expensive training and knowledge.

The following references are hereby incorporated by reference in their entirety:

(1) Dan Haught & Jim Ferguson, Microsoft Jet Database Engine Programmer's Guide, Microsoft Press, 1995

(2) Michelle A. Poolet & Michael D. Reilly, Access95 Cient/Server Development, QUE Corporation, 1996

(3) James D. Murray & William VanRyder, Encyclopedia of Graphics File Formats, O'Reilly & Associates, 1994

(4) Michael Amundsen & Curtis Smith, Teach Yourself Database Programming with Visual Basic 4 in 21 Days, Sam's Publishing, 1996

(5) Zane Thomas Et al, Visual Basic 4 How -To, The Waite Group Press, 1995

BRIEF SUMMARY OF THE INVENTION

The present invention relates to a method and apparatus for a scanning and document management system using hardware and computer software technology. More particularly, the present invention relates to an efficient methodology for scanning, generating instructions to be performed by a computer based on unique images digitized into the computer, converting to editable and searchable text, organizing and separating in electronic storage, labeling, annotating, viewing, accessing, manipulating, searching and printing of documents. A document consists of one or more pages or sheets containing text and/or graphic symbols. The pages or sheets comprising a document may be derived from any source, including scanned pages and captured video. A document may be in color, in black and white or both. The invention is useful for various commercial applications such as for example in a law practice for case litigation support imaging database to keep track of a plurality of documents produced during litigation.

The present invention comprises a scanning and document management system with a mechanism for generating instructions to be performed by a computer based on unique images digitized into the computer, which preferably includes a scanning device electronically coupled to software-enabled computer. In the preferred embodiment, the enabling software for scanning, converting to editable and searchable text, organizing and separating in electronic storage, labeling, annotating, viewing, accessing, manipulating, searching, and printing of documents is resident on

the computer. However, as one skilled in the art will appreciate, the software module for controlling the scanning device may be loaded as part of the scanning device, or scanning device and computer may be loaded with portions of the software module.

The method and apparatus for generating instructions to be performed by a computer based on unique images digitized into the computer is facilitated by a unique graphic pattern embodied on a physical medium, such as a sheet of paper or video tape. The image is electronically digitized into a computer. For example, the image on sheet of paper is digitized by scanning the sheet into the computer. Similarly, the image on a video tape may be digitized into the computer by processing the analog video signals through a commercially available hardware add-on board installed in the computer. The add-on board contains electronic circuitry capable of converting analog video signals to digital video signals. Commercially available presentation software permits viewing of the digital video on the computer monitor. Additionally, the software contains functionality to freeze a particular video frame to save as an individual digitized image. This image may contain a unique graphic pattern designed to provide one or more instructions to the computer. A special software module examines and interprets the digitized image and thus produces one or more computer instructions. The computer executes the generated instructions to produce a desired result. An example of a practical application of this invention is in the use of a physical document separator, containing a unique graphic pattern (image), to mark the beginning and end of each document in a large volume of documents scanned collectively into a computer as a single electronic file. A software module is used to examine and recognize the document separator images and thus produce and store a separate electronic file for each individual document contained in the single larger electronic file containing a plurality of documents. This is particularly desirable in various commercial applications such as in a law practice case litigation support imaging database where large volumes of documents are scanned to be electronically managed and used.

BRIEF DESCRIPTION OF THE DRAWINGS

A better understanding of the present invention can be obtained when the following detailed description of the preferred embodiment is considered in conjunction with the following drawings, in which:

Figure 1 is a block diagram of a computer system with attached accessories according to the present invention;

Figure 2 is a flowchart depicting the Document Separator logic of the preferred embodiment of the present invention;

Figure 3 is a screen shot showing the PC START screen;

Figure 4 is a screen shot showing the Main Information Screen (MIS);

Figure 5 is a screen shot of Watermark Professional Editor;

Figure 6 shows the contents of the CDINFO.DAT file;

Figure 7 is a screen shot of the Briefing Tool;

Figure 8 is a screen shot of the Transcript Viewer;

Figure 9 is a screen shot of the ISYS Query screen;

Figure 10 is a screen shot of the ISYS Search screen;

Figure 11 is a screen shot of the Word Wheel search screen;

Figure 12 is a screen shot of the System Settings screen;

Figure 13 is a screen shot of the Preferences screen;

Figure 14 is a screen shot of the Report Names screen which enables running of reports;

Figure 15 is a screen shot of the Report Names screen which enables naming of reports;

Figure 16 is a screen shot of the Duplicates Screen;

Figure 17 is a screen shot of the Exhibit List screen;

Figure 18 is a screen shot of the Litigation Bates Number Label Maker screen;

Figure 19 is a screen shot of the Litigation Document Number Label Maker screen;

Figure 20 is a screen shot of the Tempest Image Printer screen;

Figure 21 is a screen shot of Luke's Watermark Scan Utility screen;

Figure 22 depicts the Document Separator of the preferred embodiment;

Figure 23 is an example of a multi-page TIFF file;

Figure 24 shows a page divided into sixty-four sections;

Figure 25 shows further division of the sixty-four sections into four quadrants;

Figure 26 is an overlay of the 64 sections and 4 quadrants on a document page;

Figure 27 is an overlay of the 64 section and 4 quadrants on the designated Document Separator;

Figure 28 shows documents separated after execution of the Document Separator module;

Figure 29 is a screen shot of the Watermark Exhibit Scan Utility; and

Figure 30 is a screen shot of Luke's Automated OCR'ing Utility.

DETAILED DESCRIPTION OF THE INVENTION

According to the preferred embodiment, software for the present invention is developed using Microsoft's Visual Basic programming language in 32-bit mode. The software portion of the preferred embodiment uses Microsoft Jet as the database engine.

Referring first to Figure 1, an illustrative computer system 1-100 which is programmed according to the present invention and which operates according to the present invention is shown. The computer system 1-100 generally comprises a video display system 1-110, a keyboard 1-120, and a mouse 1-130. The computer system 1-100 also preferably includes various standard components, including at least one central processing unit (CPU), memory, a hard drive, a CD-ROM drive, a floppy disk drive, one or more buses, and a power supply. The computer system 1-100 of the preferred embodiment, includes a 200 Mhz Pentium MMX CPU, 32 megabytes of random access memory (RAM), 4 gigabytes of hard disk space, a 24X CD-ROM drive, 3.5" 1.4 megabyte floppy disk drive, a 17 inch monitor with 1024x768 resolution and a similarly equipped video card. In the preferred embodiment, the software program is stored on a CD-ROM disk 1-400, floppy disks 1-500 and/or hard drive of the computer 1-100 for execution by the CPU. The preferred embodiment of the present invention also includes a high-speed scanner 1-200, such as the commercially available Fujitsu scanner M3093GX, connected to the computer system 1-100. The preferred Fujitsu scanner is rated with a scanning speed of 27 pages per minute and is capable of 200 to 400 DPI resolution. Alternatively, any industry standard, i.e., TWAIN compliant, scanner with scanning speed of 24 to 30 pages per minute and 200 to 400 DPI resolution will satisfy the requirements of the preferred embodiment. The preferred embodiment uses the optional sheet feeder 1-210 of the scanner 1-200 to facilitate high speed scanning of documents into the system. The present invention preferably implements a document separator 1-300 placed at the end of each document to separately define the beginning and end of each document in a stack. A high-speed laser printer 1-600, also preferably is attached to the computer system 1-100. The preferred embodiment uses the foregoing components to practice the present invention, as described below. One skilled in the art will understand, however, that modifications or omissions may be made to the list of preferred components without departing from the principles of the present invention.

The preferred embodiment of the present invention uses software containing various

programming modules. The software portion of the preferred embodiment of the present invention is incorporated in its entirety and attached as the Appendix. Referring now to page 4 of the Appendix, the PC START module functions as the main menu for the database management system. The PC START module is an overlay program that acts as the master control program to manage a plurality of databases. In the preferred embodiment, each document database contains information on a lawsuit or legal case, although the present invention has other applications. Referring now to Figure 3, the PC START screen provides the user with options to perform various functions on each case database. For example, the user can select functions such as Load Case 3-100, Create Case 3-200, Delete Case 3-300, Repair Case 3-400 and Compress Case 3-500. Furthermore, the user can perform various system administration functions by selecting System Admin 3-600, produce labels by selecting Make Labels 3-700 or print documents by selecting Print Docs 3-800. The Current Cases list-box 3-900 permits the user to select which case database(s) will be impacted by selection of one or more of the foregoing functions.

Referring now to pages 2 to 3 of the Appendix, the Load Case module enables the user to load the selected case 3-900 into the computer system's memory 1-100. In order to load a case, the case must exist in the Current Cases list-box 3-900. The Main Information Screen (MIS), Figure 4, is displayed upon selection of the Load Case 3-100 option from the PC START screen. (Figure 3). The MIS, Figure 4, contains the document image and document briefing information. Document briefing is accomplished by the entry of summary text into various fields on the MIS such as "To" 4-110, "From" 4-120, "CC's" 4-130, "Description" 4-140 and "Comments" 4-150. Each of the foregoing briefing fields can be searched while in the MIS, Figure 4, by simply double-clicking on the desired field to search.

Case Issues 4-200 on the MIS, Figure 4, enable the user to designate special characteristics of the currently displayed document in document thumbnail 4-300. The special characteristics are designated by placing a checkmark in the box provided to the right of each Case Issue 4-200. The use of Case Issues enables a novice user to produce sophisticated reports without any programming on the part of the user. For example, the user can easily produce a report containing all documents in the case database which involve expert testimony. Pages 15 to 17 of the Appendix show the programming logic to produce the reports based on case issues defined and selected by the user. Case Issues 4-200 are user-defined and thus are not predetermined by the system. The user defines case issues relevant to a case database in the Preferences screen, Figure 13. Referring now to page 41 of the Appendix, the

Preferences screen, Figure 13, is selected by clicking on the Settings button 12-200 of the System Settings screen, Figure 12. In the Preferences screen, Figure 13, the user performs a one-time setup of chosen case issues by entering a case issue in each field of the data entry row 13-100. The case issues entered in the data entry row 13-100 are relationally connected to the Case Issues 4-200 on the MIS, Figure 4, the report screens, Figures 14 and 15, and all associated program code and tables. The Preferences screen, Figure 13, additionally permits the user to determine system folders 13-200 in which files related to a particular case database will reside on the computer system 1-100.

Referring now to page 15 of the Appendix, the document thumbnail 4-300 will be displayed in an individual window if the user double-clicks on the document thumbnail 4-300 using the mouse 1-130 or selects the Go To button 4-410. Any commercially available image-viewing program may be used to display the thumbnail document 4-300. The preferred embodiment of the present invention uses commercially available Watermark Professional Edition image viewer. While in the image viewer, Figure 5, the document display size may be changed to suit the user's needs. Furthermore, among other things, the user can print the document, annotate the document using text or audio annotations and highlight significant portions of the document.

Referring now to page 53 of the Appendix and Figure 6, the system is capable of automatically determining the location of a user-requested document in a multi-volume CD-ROM document image database. This feature is facilitated by the system's ability automatically keep track of the location of all documents as they are scanned into the system, even if more than one CD-ROM is required to store the scanned document images. This functionality is useful in circumstances where the user's computer system has a single CD-ROM drive. The functionality is particularly helpful on laptop computers, which customarily have a single CD-ROM drive.

The preferred embodiment maintains two different versions of each document in each case database: (1) the scanned image and (2) the full-text OCR version. The scanned image version is displayed in thumbnail mode 4-300 and using the image viewer program. The scanned image version is the more complete and accurate version of a document because it is essentially an identical image of the document as it was scanned into the system. While the scanned image is ideal for viewing an identical copy of the original document, the scanned image lacks the capability to be searched for text. The physical limitation of the scanned image version necessitates a full-text version of the document. The full-text version is

obtained though the use of optical character recognition (OCR) software to convert the scanned image into a text-only file (see discussion below on the OCR process).

Referring now to pages 26 to 36 of the Appendix, the Briefing Tool, Figure 7, of the preferred embodiment enables the user to view the scanned image of the document thumbnail 4-300 by clicking on the Brief button 4-420 on the MIS, Figure 4. The Briefing Tool, Figure 7, differs from the image viewer, Figure 5, in that the Briefing Tool displays the document image at all time while it is open. It is desirable to have the functionality of an image viewer that remains on top of all other windows at all times because it permits the user to have both the MIS, Figure 4, and Briefing Tool, Figure 7, up on the monitor at the same time to facilitate efficient briefing of the document in the MIS. This functionality facilitates multiple users briefing the same case database at the same time without having the need to have actual physical document copies accessible to each briefer. Additionally, the Briefing Tool reduces the likelihood of lost physical document copies. As evident from the programming source code on page 26 of the Appendix, the Briefing Tool, Figure 7, of the preferred embodiment is written in TMS/Sequia OCX technology.

Referring now to pages 13 and 69 to 83 of the Appendix, the Transcript Viewer, Figure 8, of the preferred embodiment is an ASCII transcript viewer. It displays an entire page of a deposition proceeding at a time. The Transcript Viewer is initiated by clicking on the Transcr button 4-440 on the MIS, Figure 4. The Transcript Viewer has the capability to automatically recognize various transcript types, such as different transcript types received from court reporters, and to automatically perform all necessary formatting and loading into the system. This feature is particularly desirable because state of the art systems lack this feature and thereby require significant labor to format transcripts prior to use on other state of the art document management systems. The Transcript Viewer has a Find Word 8-100 feature, which enables the user to search transcripts for particular text strings. Another desirable feature of the Transcript Viewer in this preferred embodiment is the ability to use it in conjunction with any word processing software, such as Microsoft Word or Corel's WordPerfect. The combination use enables the user to highlight any selected portion of a transcript, then click on the Copy button 8-200 in the Transcript Viewer. The copied information can then be pasted into the word processor. Significantly, the Copy 8-200 function accomplishes much more than the average cut and paste routine. Here, the Copy 8-200 function provides additional valuable information relating to the copied transcript text. Specifically, the Copy 8-200 function automatically generates document reference including

information such as the page and line number of the copied text and the name of the transcript. This automatic functionality facilitates faster deposition summaries and easier incorporation of deposition quotes in litigation pleadings. Furthermore, the Transcript Viewer allows condensed printing of transcripts – up to four pages of transcripts on a single physical printed page.

Referring now to pages 14 to 15 of the Appendix, the ISYS Query screen, Figure 9, of the preferred embodiment enables the user to perform full-text searches on the OCR version of the documents in the case database. The ISYS Query screen, Figure 9, is initiated by clicking on the FullText button 4-470 on the MIS, Figure 4. The preferred embodiment of the present invention uses a commercially available search engine called ISYS, produced by Odyssey Development Corporation, for performing the full-text searches. The third party ISYS Search screen, Figure 10, is initiated by clicking on the Q button 9-100. The user enters queries in the query field 10-100 using search connectors 10-200 as needed. The user may choose to use the Word Wheel icon 10-300 to search for the number of occurrences of any given specific word. The Word Wheel Search screen, Figure 11, is initiated when the user clicks on the Word Wheel icon 10-300 on the ISYS Search screen, Figure 10. The user enters the specific word to search for in the query field 11-100 in Figure 11. The user must select one of the search methods 11-200. The self-explanatory choices are "Starts with" and "Sounds like." The result of the single-word Word Wheel search is displayed in the result list-box 11-300.

Referring now to page 12 of the Appendix, the Add button 4-450 and the Delete button 4-460 on the MIS, Figure 4, of the preferred embodiment enable the user to perform standard add and delete processing of case database records.

Referring now to pages 13 and 20 to 20 of the Appendix and pages 42 to 52 of the Appendix, the Reports button 4-480 on the MIS, Figure 4, enables the user to print reports as defined by the user's System Settings, Figure 12, Reports button 12-300 and Rpt Names button 12-400. Report Screen, Figure 15, enables the user to name reports. Report Screen, Figure 14, enables the user to run reports.

Referring now to page 13 of the Appendix, the Print Scr button 4-490 on the MIS, Figure 4, of the preferred embodiment provides the functionality to produce an image of an index card that looks exactly like the MIS, Figure 4.

Referring now to page 37 of the Appendix, the Duplicates button 12-500 on the System Settings screen, Figure 12, performs a check for duplicate records in the case database. The result of the duplicates check is displayed on the Duplicates Screen, Figure 16.

The Duplicates Screen is a grid type screen that includes an entry for every document in the database. The duplicates checking routine performs approximately 14 separate checks on document numbers. Duplicate records are grouped together based on document numbers. Duplicate records are highlighted and deleted by pressing the delete key. The Duplicates Screen has the added functionality of providing the database administrator the capability to perform mass maintenance functions on all records in the database. For example, the administrator can globally make changes to all data in a case database. Various reports can also be produced using the Duplicates Screen. The reporting capabilities found here, i.e., Figure 16, are separate and distinct from other reporting features of the system. All appearance changes on this screen, Figure 16, such as hiding various columns for printing purposes, are temporary, while all data content changes, such as global find and replace, are permanent.

Referring now to page 37 of the Appendix, the Exhibit List button 12-600 on the System Settings screen, Figure 12, displays the Exhibit List screen, Figure 17. The Exhibit List screen, Figure 17, allows the user to modify parts of the data for use in an automatic exhibit list which can be printed from the Report Screens, Figures 14 and 15.

Referring now to pages 14 to 15 of the Appendix, the preferred embodiment of the present invention provides the user with the option of selecting the user's choice of full-text search engine Figure 9. The system of the preferred embodiment is designed to automatically detect what particular component is installed on the user's computer system 1-100 and to automatically develop necessary links to enable use of the installed components as the user's full-text search engine.

Referring now to pages 1 to 2 of the Appendix, the Create Case module permits the user to add a new case database. A case database must be added to the computer system 1-100 before any of the database actions listed in Figure 3 can be performed on the documents and the associated information which comprise a case database. A series of dialogue boxes step the user through the process of creating a new database. The procedure for adding a new case database is initiated by clicking on the Create Case button 3-200 on the PC START screen, Figure 3.

Referring now to page 2 of the Appendix, the Delete Case module permits the user to delete a case database from the computer system 1-100. In order to delete a case, the case must exist in the Current Cases list-box 3-900 and must be selected by the user. The delete case function is password protected to prevent unauthorized deletion of case databases. A

series of dialogue boxes step the user through the process of deleting a database. The procedure for deleting a case database is initiated by clicking on the Delete Case button 3-300 on the PC START screen, Figure 3.

Referring now to pages 3 to 4 of the Appendix, the Repair Case module permits the user to perform various repair functions on a case database residing on the computer system 1-100. In order to repair a case, the case must exist in the Current Cases list-box 3-900 and must be selected by the user. The Repair Case module corrects technical problems with the case database such as damaged or defective files. It is critical to repair a case when a damaged or defective file prevents the system from functioning as intended. A series of dialogue boxes step the user through the process of repairing a database. The procedure for repairing a case database is initiated by clicking on the Repair Case button 3-400 on the PC START screen, Figure 3.

Referring now to page 1 of the Appendix, the Compress Case module permits the user to perform various maintenance functions on a case database residing on the computer system 1-100. In order to compress a case, the case must exist in the Current Cases list-box 3-900 and must be selected by the user. The Compress Case module performs tasks such as reorganization and defragmentation of the selected case database. During the normal course of use, case databases are added and deleted as needed. However, the system does not physically delete a case database and its associated files until the case database is compressed. Thus, it is desirable to compress a case to free up storage space occupied logically deleted files. Compressing a case improves overall performance of the system and allows faster processing of information contained in the case database. A series of dialogue boxes step the user through the process of compressing a database. The procedure for compressing a case database is initiated by clicking on the Compress Case button 3-500 on the PC START screen, Figure 3.

Referring now to page 8 and 9 of the Appendix, the Make Labels module allows the user to print litigation label numbers based on either the Bates numbers stamped document production during litigation or case database system specific Document Number. Figure 18 demonstrates the preferred embodiment's form for Bates number based label maker and Figure 19 demonstrates the preferred embodiment's form for Document number based label maker.

Referring now to pages 5 and 84 to 87 of the Appendix, the Print Docs module enables the user to perform high-speed document production. The module contains functionality which acts as a batch print utility for delayed or scheduled printing of scanned image files on

the high-speed laser printer 1-600. The batch print capability of the Print Docs module is very useful as a cost-effective method for automated and scheduled high-speed document production. Referring now to Figure 20, the Tempest Image Printer screen enables the user to select one or more files to be printed. By simply clicking on check boxes, the system is capable of batching all files for deferred printing and production of selected documents on high-speed network printers. This feature eliminates costs associated with manual production of documents by clerks – a process that requires clerks to manually locate and copy selected documents for production. Current state of the art technology does not provide batch print capability in document management systems. Further performance enhancement may be realized by installing additional memory chips into the high-speed laser printer 1-600. The preferred embodiment of the present invention uses the commercially available Hewlett Packard 5Si model. However, the preferred embodiment uses a 5Si model with its standard memory upgraded to 32 megabytes. The upgraded 5Si reduces the processing load on the computer system's 1-100 resources and effectively improves overall system performance. The present invention envisions a system whereby the batch print functionality may be easily expanded by one skilled in the art to enable multiple printers to simultaneously share the burden of printing high-volume batched documents. The accelerated print management produces significant time and labor savings.

Referring now to pages 54 to 60 of the Appendix, Luke's Watermark Scan Utility module is the mechanism of the preferred embodiment of the present invention by which all documents that belong to a case database are scanned into the computer system 1-100. The documents are scanned into the computer system 1-100 using the attached high-speed scanner 1-200. Luke's Watermark Scan Utility screen, Figure 21, is displayed when the scanning utility module is initiated. The scanning utility of the preferred embodiment will work with any industry standard, i.e., TWAIN compliant, scanner. However, the utility can be modified to work with specific high-speed scanners. For example, the scan utility of the preferred embodiment includes special driver software for the preferred Fujitsu high-speed scanner. Scanner setup is accomplished by means of clicking on the Setup button 21-100 in Figure 21. The scan utility automatically links the scanned image into the case database and gives it a document number and bates label number. The automatic linking and setup of the scanned images into the case database is highly desirable because it saves significant manual effort that would otherwise be required. The Scan button 21-200 initiates scanning of as many documents as are in the scanner's 1-200 sheet feeder 1-210. The Single button 21-300

initiates scanning of only a single page of a document. The Scan90 button 21-400 makes a 90-degree orientation adjustment to a document as it is scanned into the system. This eliminates the need for the user to adjust the scanned image when viewing it. The Save button 21-500 permanently stores scanner settings. The scan utility keeps a running tab of all scanned images that have not yet been processed by the OCR routine, i.e., conversion to full-text version of the scanned image. This is desirable because the OCR list generated by the scan utility is used in the OCR process without having to manually determine which scanned images need to be OCRed.

Referring now to pages 94 to 98 of the Appendix, the Document Separator module is designed to facilitate efficient scanning of large volumes of documents in a continuous stream. Documents are typically scanned one document at a time in order to signal to the system the end of one document and the start of another. Scanning one document at a time is a laborious and time-consuming manual procedure. It is desirable to perform continuous scanning of documents because it eliminates the need to manually signal the end of each document (containing one or more physical pages) and the start of the next document. In the preferred embodiment, continuous scanning of documents is achieved by placing a physical document separator between each of the documents before scanning. In the preferred embodiment the physical document separator, Figure 22, is a physical sheet of paper that contains a predetermined unique image pattern. All documents are then continuously scanned and saved as one multi-page electronic file, such as a TIFF file. Figure 23 shows an example of a continuous stream of documents separated by the designated uniquely patterned document separator. The document separator module examines and processes each multi-page TIFF file to produce individual documents and saves each such document as a separate electronic file. The logic of Document Separator, as used in the preferred embodiment, is described in the flowchart in Figure 2. Specifically, the process begins by loading and displaying each scanned image on the computer screen. The width and height of the scanned sheet is determined and the sheet is divided into sixty-four (64) sections, Figure 24. The 64 sections are further divided into four (4) quadrants, Figure 25. Figure 26 demonstrates overlay of the 64 sections and 4 quadrants on a page of a document. Figure 27 demonstrates overlay of the 64 sections and 4 quadrants on a page that is the designated document separator with the predetermined unique graphic pattern. Thereafter, the software module takes one thousand (1000) color samples or pixel values from each quadrant. Each color sample is examined and compared against the predetermined unique graphic pattern of the designated document separator to determine whether the scanned page is

the designated document separator. If the scanned page does not correspond identically/substantially to that of the predetermined unique image, (the Document Separator template), the scanned page is marked as part of an ongoing document and the process is repeated again. However, if the page is the designated document separator, then the end of the current document is indicated and all scanned pages prior to the document separator are saved as a single document in a separate electronic file. Figure 28 displays the end result of the foregoing process for the example multi-page TIFF file shown in Figure 23. At the completion of the document separator routine, four documents are extracted in Figure 28 from the single multi-page TIFF file in Figure 23. This process is repeated until all documents have been separated in a similar manner. The utility of the present invention lies in part in time savings realized through continuous scanning of large volumes of plurality of documents using a high-speed scanner.

Referring now to pages 61 to 66 of the Appendix, the Watermark Exhibit Scan Utility module is an automatic trial exhibit maker. It allows exhibit descriptions to be added as the documents are being scanned.

Referring now to pages 67 to 68 of the Appendix, Luke's Automated OCR'ing Utility module enables the OCR software, such as Omni Page Pro, to run in a batch file mode at various scheduled times. It corrects errors and loads the assigned documents to the OCR program.

Although the method and apparatus of the present invention has been described in connection with the preferred embodiment, it is not intended to be limited to the specific form set forth herein, but on the contrary, it is intended to cover such alternatives, modifications, and equivalents, as can be reasonably included within the spirit and scope of the invention as defined by the appended claims.

PROGRAM CODE

SEE APPENDIX 1, PAGES 1 – 98

```
Form1 - 1

' PC START
' Copyright (C) 1995 - 1997, Luke A. Spence & Michael L. Saltsman
' Last Modified on 05/9/97
' All rights reserved.  No portion of this program code may be altered,
' reproduced, used without written permission of the authors
' Compile with 16 bit processor "GetModuleUsage" API does not work with
' WIN32 or Windows NT

Option Explicit
Dim Drive Path As String, CaseFileName As String
Private Declare Function GetModuleUsage% Lib "Kernel" (ByVal hModule%)
Private Sub cmdCompress Click()
Dim Retval As Integer, CaseToCompress As String, Compressed As String
If Security() = 0 Then Exit Sub
    Retval = MsgBox("Do you want this case is to be compressed?", 275, "Case Com
press")
    If Retval = 6 Then
        CaseToCompress = Drive Path & List1.Text & ".mdb"
        Compressed = Drive Path & "PCTemp.mdb"
        CompactDatabase CaseToCompress, Compressed
        Kill CaseToCompress
        Name Compressed As Drive Path & List1.Text & ".mdb"
        MsgBox "The case is compressed.", 0, "Compressed Case"
    Else
        Exit Sub
    End If
    File1.Refresh
    UpdateList
    UpdateButtons
End Sub

Private Sub cmdCreateCase Click()
If Security() = 0 Then Exit Sub
Dim Y As Integer, Z As Integer, NewCaseName As String, OriginalFile As String
Dim Ansr, CurDrv, Msg, TmpPath, HomeDir, ChrName As String, Characters As String
Dim NewFile As String, TestFile As String
    NewCaseName = InputBox$("Please type your case name.  It can be up to eight
letters long, you may use any letter(s), number(s), or the dash '-' & /or the un
derscore '_' symbols in your description.", "Case Name", "")
    If NewCaseName = "" Then Exit Sub
    If UCase(NewCaseName) = "DOCONTR" Then
        MsgBox "Please use another filename.", 0, "Reserved Case Name."
        Exit Sub
    End If
    NewCaseName = Left$(NewCaseName, 8)
    Z = Len(NewCaseName)
    For Y = 1 To Z
       ChrName = Mid$(NewCaseName, Y, 1)
       Select Case UCase(ChrName)
       Case "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N"
  "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z"
            ChrName = ChrName
       Case "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "-", "_"
            ChrName = ChrName
       Case Else
            ChrName = ""
       End Select
       Characters = Characters + ChrName
    Next Y
    NewCaseName = Characters
    OriginalFile = Drive Path & "docontr.mdb"
    NewFile = Drive Path & NewCaseName & ".mdb"
    TestFile = Dir(NewFile)
    If TestFile = "" Then
        FileCopy OriginalFile, NewFile
    Else
        MsgBox "That case already exists. "
        File1.Refresh
        UpdateList
```

Form1 - 2

```
        UpdateButtons
        Exit Sub
    End If

    On Error Resume Next     ' Set up error handler.
    CurDrv = Left(CurDir, 2)     ' Get current drive letter.
    HomeDir = CurDir
    TmpPath = UCase(HomeDir & "\" & NewCaseName)    ' Make path specification.
    MkDir TmpPath    ' Make new directory.
    TmpPath = UCase(HomeDir & "\" & NewCaseName & "\IMAGES")
    MkDir TmpPath    ' Make new directory.
    TmpPath = UCase(HomeDir & "\" & NewCaseName & "\DOCS")
    MkDir TmpPath    ' Make new directory.
    TmpPath = UCase(HomeDir & "\" & NewCaseName & "\DEPOS")
    MkDir TmpPath    ' Make new directory.
    File1.Refresh
    UpdateList
    UpdateButtons
End Sub
Private Sub cmdDeleteCase Click()
Dim CurDrv, Msg, TmpPath, FileKill, HomeDir, caseToDelete
Dim Retval As Integer, ToDelete As String
If Security() = 0 Then Exit Sub
    Retval = MsgBox("Are you positive that this case is to be deleted?", 275, "C
ase Deletion")
    If Retval = 6 Then
        ToDelete = Drive Path & List1.Text & ".mdb"
        If InStr(List1.Text, "DOCONTR") Then Exit Sub
        Kill ToDelete

    On Error Resume Next     ' Set up error handler.
    CurDrv = Left(CurDir, 2)     ' Get current drive letter.
    HomeDir = CurDir
    FileKill = "*.*"
    caseToDelete = Drive Path & List1.Text
    TmpPath = UCase(HomeDir & "\" & caseToDelete & "\IMAGES")
    ChDir TmpPath
    Kill FileKill
    ChDir HomeDir
    RmDir TmpPath
    TmpPath = UCase(HomeDir & "\" & caseToDelete & "\DOCS")
    ChDir TmpPath
    Kill FileKill
    ChDir HomeDir
    RmDir TmpPath
    TmpPath = UCase(HomeDir & "\" & caseToDelete & "\DEPOS")
    ChDir TmpPath
    Kill FileKill
    ChDir HomeDir
    RmDir TmpPath
    TmpPath = UCase(HomeDir & "\" & caseToDelete)
    RmDir TmpPath
    Else
        Exit Sub
    End If
    MsgBox "Per your request, the case is deleted.", 0, "Deleting Case"
    File1.Refresh
    UpdateList
    UpdateButtons
End Sub
Private Sub cmdExit_Click()
    End
End Sub
Private Sub cmdLoadCase Click()
Dim L As Integer, Z As Integer, Database As String, LoadAccess As String
    On Error Resume Next
    Database = List1.Text & ".mdb"
    'a hard map was added to make this work on 97 laptops
    LoadAccess = "Chaser.exe " & "C:\PCHASER\" & Database
```

```
Form1 - 3

    L = Shell(LoadAccess, 1)
    Form1.WindowState = 1 'Minimized
    While GetModuleUsage(L) > 0
      Z = DoEvents()
    Wend
    Form1.WindowState = 0
End Sub
Private Sub cmdMinimize_Click()
    Form1.WindowState = 1
End Sub
Private Sub CmdMklbl_Click()
Dim X As Integer, Z As Integer
On Error Resume Next
X = Shell("LABEL.EXE", 1)
  Form1.WindowState = 1 'Minimized
    While GetModuleUsage(X) > 0
      Z% = DoEvents()
    Wend
    Form1.WindowState = 0
End Sub
Private Sub cmdRename_Click()
Dim Z As Integer, Y As Integer, NewCaseName As String, ChrName As String, Charac
ters As String
Dim OriginalFile As String, NewFile As String, TestFile As String
If Security() = 0 Then Exit Sub
Dim dirToChange, dirToChangeTo
    NewCaseName = InputBox$("Please type your case name.  It can be up to eight
letters long, you may use any letter(s), number(s), or the dash '-' & /or the un
derscore '_' symbols in your description.", "Case Name", "")
    If NewCaseName = "" Then Exit Sub
    If UCase(NewCaseName) = "DOCONTR" Then
        MsgBox "Please use another filename.", 0, "Reserved Case Name."
        Exit Sub
    End If
    NewCaseName = Left$(NewCaseName, 8)
    Z = Len(NewCaseName)
    For Y = 1 To Z
       ChrName = Mid$(NewCaseName, Y, 1)
       Select Case UCase(ChrName)
       Case "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N"
  "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z"
            ChrName = ChrName
       Case "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "-", "_"
            ChrName = ChrName
       Case Else
            ChrName = ""
       End Select
       Characters = Characters + ChrName
    Next Y
    NewCaseName = Characters
    OriginalFile = Drive Path & List1.Text & ".mdb"
    NewFile = Drive Path & NewCaseName & ".mdb"
    dirToChange = CurDir & "\" & List1.Text & "\"
    dirToChangeTo = CurDir & "\" & NewCaseName & "\"
    TestFile = Dir(NewFile)
    If TestFile = "" Then
        Name OriginalFile As NewFile
        MsgBox dirToChange
        MsgBox dirToChangeTo
        Name dirToChange As dirToChangeTo
    Else
        MsgBox "That case already exists."
        Exit Sub
    End If
    File1.Refresh
    UpdateList
    UpdateButtons
End Sub
Private Sub cmdRepair_Click()
```

```
Form1 - 4

Dim Retval As Integer, CaseToRepair As String
If Security() = 0 Then Exit Sub
    Retval = MsgBox("Do you want this case is to be repaired?", 275, "Case Repai
r")
    If Retval = 6 Then
        CaseToRepair = Drive Path & List1.Text & ".mdb"
        RepairDatabase CaseToRepair
        MsgBox "The case is repaired.", 0, "Repaired Case"
    Else
        Exit Sub
    End If
    File1.Refresh
    UpdateList
    UpdateButtons
End Sub
Private Sub Command1_Click()
  End
End Sub
Private Sub Dir1 Change()
    File1.Path = Dir1.Path
End Sub
Private Sub Drive1 Change()
    On Error Resume Next
    Dir1.Path = Drive1.Drive
End Sub
Private Sub Form Load()
    On Error Resume Next
    ' center form
    Me.Left = (Screen.Width - Me.Width) / 2
    Me.Top = (Screen.Height - Me.Height) / 2
    pctLogo.Picture = LoadPicture("backgrnd.bmp")
    UpdateList
    If Dir(Drive Path & "DOCONTR.MDB") = "" Then cmdCreateCase.Enabled = False
    cmdDeleteCase.Enabled = False
    cmdLoadCase.Enabled = False
    cmdRename.Enabled = False
    cmdRepair.Enabled = False
    cmdCompress.Enabled = False
    cmdSysAdmin.Enabled = False
End Sub
Private Sub List1 Click()
     If List1.Text = "DOCONTR" Then
        cmdDeleteCase.Enabled = False
        cmdRename.Enabled = False
     Else
        cmdDeleteCase.Enabled = True
        cmdRename.Enabled = True
        cmdRepair.Enabled = True
        cmdCompress.Enabled = True
     End If
     cmdSysAdmin.Enabled = True
     cmdLoadCase.Enabled = True
End Sub
Private Sub List1 DblClick()
  cmdLoadCase.Value = True
  Rem cmdSysAdmin.Value = True
End Sub
Private Function Security()
Dim Ans
Security = 0
Ans = InputBox("Enter your Paper Chaser Administration Password", "Security")
If Ans = "Rufus" Or Ans = "rufus" Then
    Security = 1
    Else
    MsgBox "Access Prohibited.  Incorrect Password", 16, "Security Check"
    End If
End Function
Private Sub UpdateButtons()
    cmdLoadCase.Enabled = False
```

Form1 - 5

```
    cmdDeleteCase.Enabled = False
    cmdRename.Enabled = False
    cmdRepair.Enabled = False
    cmdCompress.Enabled = False
    cmdSysAdmin.Enabled = False
End Sub
Private Sub UpdateList()
Dim X As Integer, I As Integer, CaseName As String, Test As String
    List1.Clear
    For I = 0 To File1.ListCount - 1
        CaseFileName = File1.List(I)
        For X = 1 To 8
            Test = Mid$(CaseFileName, X, 1)
            If Test = "." Then Exit For
            CaseName = CaseName + UCase(Test)
        Next X
        List1.AddItem CaseName
        CaseName = ""
    Next I
End Sub
Private Sub CmdSysAdmin_Click()
Dim X As Integer, Z As Integer, LoadAccess As String, Database As String
    On Error Resume Next
    Database = List1.Text & ".mdb"
    LoadAccess = "msarn200.exe " & Database & " /ini pchaser.ini"
    X = Shell(LoadAccess, 1)
      Form1.WindowState = 1 'Minimized
    While GetModuleUsage(X) > 0
      Z% = DoEvents()
    Wend
    Form1.WindowState = 0
End Sub
Private Sub PrintDocs_Click()
Dim X As Integer
On Error Resume Next
X = Shell("TIFFPRINT.EXE", 1)
Rem Form1.WindowState = 1 'Minimized
Rem   While GetModuleUsage(X) > 0
Rem   Z = DoEvents()
Rem   Wend
Rem   Form1.WindowState = 0
End Sub
```

```
frmStartUP - 1

Private Sub Form Click()
    Timer1.Enabled = False
    Load Form1
    Form1.Show
    Unload frmStartUP
End Sub

Private Sub Form Load()
    On Error Resume Next
    If App.PrevInstance Then
      Beep
      End
    End If
    frmStartUP.Show
   Img1.Picture = LoadPicture("start-up.bmp")
   'Img1.Left = (Screen.Width - Img1.Width) / 2
   'Img1.Top = (Screen.Height - Img1.Height) / 2
    Me.Left = (Screen.Width - Me.Width) / 2
    Me.Top = (Screen.Height - Me.Height) / 2
End Sub

Private Sub Timer1 Timer()
    Timer1.Enabled = False
    Load Form1
    Form1.Show
    Unload frmStartUP
End Sub
```

```
Form1 - 1

' Label Maker Copyright (C) 1995, 1996  Luke A. Spence
' Last modified on 01/02/96
' All rights reserved.  No portion of this cprogram code may be altered,
' reproduced or used without written permission of the authors.
Option Explicit
DefInt A-Z

Dim paddedSuffix$
Dim paddedPrefix$
Dim prefixLength%
Dim suffixLength%
Dim startNumberLength%
Dim adjusted%
Dim page%
Dim pages%
Dim lineNumber%
Dim column%
Dim labelsPrinted%
Dim skip%
Dim skipLine%
Dim startLineNumber%
Dim startColumnNumber%
Dim bateStampNumber&

Private Sub cmdExit_Click()
    End
End Sub

Private Sub cmdHelp_Click()
  frmHelp.Left = (Screen.Width - frmHelp.Width) / 2
  frmHelp.Top = (Screen.Height - frmHelp.Height) / 2
  Load frmHelp
  frmHelp.Show 1
End Sub

Private Sub cmdPrint_Click()
    PrintLabels
End Sub

Private Sub DetermineCenter()

  If Len(txtPrefix) = 0 Then          'Adding a space to the end of
    paddedPrefix = ""                 'the prefix, if it exists, to
  Else                                'keep it and the bateStampNumber
    paddedPrefix = txtPrefix & " "    'one space apart on the printout
  End If
  prefixLength = Len(paddedPrefix)

  If Len(txtSuffix) = 0 Then          'Adding a space to the
    paddedSuffix = ""                 'beginning of the suffix, if
  Else                                'it exists, to keep it and
    paddedSuffix = " " & txtSuffix    'the bateStampNumber one
  End If                              'space apart on the printout
  suffixLength = Len(paddedSuffix)

  If optNoSpacing = True Then
    startNumberLength = Len(txtStartNumber)
  Else
    startNumberLength = 7
  End If

  ' the following determines the center
  adjusted = 8 - ((prefixLength + startNumberLength + suffixLength) \ 2)

End Sub

Private Sub Form_Load()
  cmdPrint.Enabled = False
```

```
Form1 - 2

End Sub

Private Sub optNumOfLabels_Click()
  txtNumRequired = ""
End Sub

Private Sub optNumOfPages_Click()
  txtNumRequired = ""
End Sub

Private Sub optPrefixNO_Click(Value As Integer)
    UpdateLength
End Sub

Private Sub optPrefixYES_Click(Value As Integer)
    UpdateLength
End Sub

Private Sub optsuffixNO_Click(Value As Integer)
    UpdateLength
End Sub

Private Sub optsuffixYES_Click(Value As Integer)
    UpdateLength
End Sub

Private Sub PrintLabels()
  pnlDisplay.Caption = "Printing..."
  On Error Resume Next
  If txtStartAtColumn = "" Then txtStartAtColumn = 1
  If txtStartAtRow = "" Then txtStartAtRow = 1
  startLineNumber = txtStartAtRow
  startColumnNumber = txtStartAtColumn

  DetermineCenter

  Printer.FontName = "Courier"
  Printer.FontSize = 12
  Printer.FontBold = True

  labelsPrinted = 0
  bateStampNumber& = txtStartNumber
  pages = lblNumOfPages + 1

  For page = 1 To pages
    For lineNumber = startLineNumber To 20
      If startLineNumber > 1 Then
        For skipLine = 1 To startLineNumber - 1
          Printer.Print "": Printer.Print "": Printer.Print ""
        Next skipLine
      End If
      startLineNumber = 1
      Printer.Print "": Printer.Print "". Printer.Print ""
      For column = startColumnNumber To 4
        startColumnNumber = 1
        If Int(labelsPrinted) >= Int(lblNumOfLabels) Then GoTo Done
        If column = 1 Then skip = adjusted + 2
        If column = 2 Then skip = adjusted + 23
        If column = 3 Then skip = adjusted + 43
        If column = 4 Then skip = adjusted + 63
        Printer.Print Tab(skip);
        Printer.Print paddedPrefix;
        If optZeros = True Then
          Printer.Print Format$(bateStampNumber&, "0000000");
        ElseIf optSpaces = True Then
          Printer.Print Format$(bateStampNumber&, "@@@@@@@");
        Else
          Printer.Print Format$(bateStampNumber&, "#######");
        End If
```

```
Form1 - 3

        Printer.Print paddedSuffix;
        If bateStampNumber& = 9999999 Then
          column = 4
          lineNumber = 20
          page = pages
        End If
        bateStampNumber& = bateStampNumber& + 1
        labelsPrinted = labelsPrinted + 1
        skip = skip + 20     ' adds 20 spaces to the 'Tab'
      Next column
    Next lineNumber
    Printer.NewPage
  Next page
  Printer.EndDoc
  Exit Sub

Done:
  Printer.NewPage
  Printer.EndDoc
  pnlDisplay.Caption = "Print job sent to the printer."
  Exit Sub

End Sub

Private Sub txtNumRequired_Change()
  On Error Resume Next
  If optNumOfPages = True Then
    lblNumOfLabels = txtNumRequired * 80
    lblNumOfPages = txtNumRequired
  End If
  If optNumOfLabels = True Then
    lblNumOfLabels = txtNumRequired
    lblNumOfPages = Format$((txtNumRequired / 80), "####.##")
  End If
  If txtNumRequired = "" Then
    lblNumOfLabels = ""
    lblNumOfPages = ""
  End If
  If txtStartNumber <> "" And txtNumRequired <> "" Then
    cmdPrint.Enabled = True
  Else
    cmdPrint.Enabled = False
  End If
End Sub

Private Sub txtStartAtColumn_Change()
  On Error Resume Next
  If txtStartAtColumn.Text < 1 Then txtStartAtColumn.Text = ""
  If txtStartAtColumn.Text > 4 Then txtStartAtColumn.Text = ""
End Sub

Private Sub txtStartAtRow_Change()
  On Error Resume Next
  If txtStartAtRow.Text < 1 Then txtStartAtRow.Text = ""
  If txtStartAtRow.Text > 20 Then txtStartAtRow.Text = ""
End Sub

Private Sub txtStartNumber_Change()
  If txtStartNumber <> "" And txtNumRequired <> "" Then
    cmdPrint.Enabled = True
  Else
    cmdPrint.Enabled = False
  End If
End Sub

Private Sub txtStartNumber_KeyPress(KeyAscii As Integer)
  Select Case KeyAscii
  Case 48 To 57
    ' do nothing, valid entry
```

```
Form1 - 4

  Case 8
    ' backspace character
  Case Else
    KeyAscii = 0 '48
  End Select
End Sub

Private Sub UpdateLength()
   If optPrefixYES = True Then
      If optSuffixYes = True Then      ' SUFFIX & PREFIX
         'PREFIX
         txtPrefix.MaxLength = 3
         txtPrefix.Left = 480
         txtPrefix.Width = 615
         txtPrefix.Visible = True
         lblPrefix.Left = 480
         lblPrefix.Visible = True
         'SUFFIX
         txtSuffix.MaxLength = 3
         txtSuffix.Left = 3120
         txtSuffix.Width = 615
         txtSuffix.Visible = True
         lblSuffix.Left = 3120
         lblSuffix.Visible = True
         'NUMBER
         txtStartNumber.Left = 1440
         lblStartNumber.Left = 1440
      Else
         'SUFFIX
         txtSuffix.Visible = False     ' PREFIX
         lblSuffix.Visible = False
         txtSuffix = ""
         'PREFIX
         txtPrefix.MaxLength = 7
         txtPrefix.Left = 600
         txtPrefix.Width = 1300
         txtPrefix.Visible = True
         lblPrefix.Left = 600
         lblPrefix.Visible = True
         'NUMBER
         txtStartNumber.Left = 2300
         lblStartNumber.Left = 2300
      End If
   ElseIf optSuffixYes Then            ' SUFFIX
      txtPrefix.Visible = False
      lblPrefix.Visible = False
      txtPrefix = ""
      'SUFFIX
      txtSuffix.MaxLength = 7
      txtSuffix.Left = 2300
      txtSuffix.Width = 1300
      txtSuffix.Visible = True
      lblSuffix.Left = 2300
      lblSuffix.Visible = True
      'NUMBER
      txtStartNumber.Left = 600
      lblStartNumber.Left = 600
   Else                               ' NONE
      txtPrefix.Visible = False
      lblPrefix.Visible = False
      txtPrefix = ""
      txtSuffix.Visible = False
      lblSuffix.Visible = False
      txtSuffix = ""
      txtStartNumber.Left = 1400
      lblStartNumber.Left = 1400
   End If
End Sub
```

```
frmStartUp - 1

' Copyright (C) 1995,19[illegible] Luke Spence
' Last modified on 01/02/96

Private Sub Form Click()
  Timer1.Enabled = False
  Load Form1
  Form1.Show
  Unload frmStartUp
End Sub

Private Sub Form Load()
  frmStartUp.Left = (Screen.Width - frmStartUp.Width) / 2
  frmStartUp.Top = (Screen.Height - frmStartUp.Height) / 2
  frmStartUp.Show
  Form1.Left = (Screen.Width - Form1.Width) / 2
  Form1.Top = (Screen.Height - Form1.Height) / 2
  Load Form1
End Sub

Private Sub Timer1_Timer()
  Load Form1
  Form1.Show
  Unload frmStartUp
End Sub


frmHelp - 1

' Copyright (C) 1995,1996  Luke Spence
' Last modified on 01/02/96

Option Explicit

Private Sub cmdReturn_Click()
  Unload frmHelp
End Sub
```

```
Form1 - 1

'Paper Chaser
'Copyright(C) 1995 - 1997, Michael L. Saltsman
'All rights reserved.  No portion of this program code may be altered,used or re
produced
'without written permission of the author.
Option Explicit
' declare API functions
Private Declare Function GetDriveType Lib "kernel32" Alias "GetDriveTypeA" (ByVa
l nDrive As String) As Long
' declare API constants
Const DRIVE_CDROM = 5
Const DRIVE_FIXED = 3
Const DRIVE_RAMDISK = 6
Const DRIVE_REMOTE = 4
Const DRIVE_REMOVABLE = 2

Public DriveType As String
Public CurrentDisk As Variant
Public DiskNum As Variant
Dim CDROM As String
Public MyCriteria As String
Dim ImagePath As String
Public NoC As Variant
Dim StrBuffer As String * 250
Private Sub CmdAdd_Click()
Data1.Recordset.AddNew
Data1.Recordset.Update
Data1.Recordset.MoveLast
Text34.Text = "Q.TIF"
End Sub
Private Sub CmdBrief_Click()
Dim Temp As String, R As Double, Test As String
Label34.DataField = "ImagePath"
Temp = Label34.Caption & Text34.Text
Test = Label34.Caption & Text34.Text
If Dir(Test) <> "" Then
    R = Shell("DOCIV.EXE " & Temp, 1)
    Exit Sub
    Else
    MsgBox "Image not available. Wrong disk.  Use Go To to re-enter document num
ber to find disk number."
    End If
End Sub

Private Sub CmdDelete_Click()
Dim Msg As String, Title As String, Style As Variant, Response As Variant, MyStr
ing As String
Msg = "Delete this entry?"    ' Define message.
Style = vbYesNo + vbQuestion + vbDefaultButton2 ' Define buttons.
Title = "Delete Entry Confirmation"  ' Define title.
Response = MsgBox(Msg, Style, Title)
If Response = vbYes Then     ' User chose Yes.
    MyString = "Yes"     ' Perform some action.
    Data1.Recordset.Delete
    Data1.Refresh
    Data2.Refresh
    Exit Sub
Else     ' User chose No.
    MyString = "No" ' Perform some action.
    Beep
    Exit Sub
End If
End Sub
Private Sub CmdEnd_Click()
End
End Sub
Private Sub CmdGoto_Click()
Dim MyCriteria As String
FrmSearch.Label11.Caption = "Enter the Document # to find:"
```

```
Form1 - 2

gs FieldName$ = "[Doc Number]"
Load GotoDoc
GotoDoc.Show
End Sub
Private Sub CmdPrintScreen_Click()
Dim Msg
On Error GoTo ErrorHandler
Form1.PrintForm
Printer.EndDoc
Exit Sub
ErrorHandler:
    Msg = "The form can't be printed."
    MsgBox Msg  ' Display message.
    Resume Next
End Sub
Private Sub CmdRpts_Click()
Dim LoadAccess As String, X As Variant
Me.Left = (Screen.Width - Me.Width) / 2
Me.Top = (Screen.Height - Me.Height) / 2
Rem Load FrmRpt
Rem FrmRpt.Show
Rem ProgramPath.DataField = "DbPath"
Rem MsgBox gs FileName$
Rem REMOVE THIS HARD CODE
LoadAccess = "C:\PCHASER\msarn200.exe " & " C:\PCHASER\" & gs_FileName$ & " /ini
 pchaser.ini"
    X = Shell(LoadAccess, 1)
End Sub
Private Sub CmdTrans_Click()
Dim R As Double, Temp As String
Label34.DataField = "ImagePath"
Temp = Label34.Caption & Text34.Text
R = Shell("SUMCLONE.EXE " & Temp, 1)
End Sub

Private Sub Form_Load()
Dim MyDb As String, MyDrive As String, Z As Integer
Rem PLACED HERE BECUASE OF PATH PROBLEMS ON CDROM LOAD UP
On Error Resume Next
' hourglass cursor
MousePointer = 11
Me.Left = (Screen.Width - Me.Width) / 2
Me.Top = (Screen.Height - Me.Height) / 2
gs FileName$ = Command$
Rem MsgBox Command$
Data1.DatabaseName = gs FileName$
Data2.DatabaseName = gs FileName$
Data3.DatabaseName = gs FileName$
Data1.RecordSource = "Select * from [Document Info]order by [Doc Number]"
Data2.RecordSource = "Select * from [Preferences]"
Data3.RecordSource = "Select * from [Disk Names]"
Text1.DataField = "Doc Number"
Text2.DataField = "Doc Date"
Text3.DataField = "Beg Bates #"
Text4.DataField = "End Bates #"
Text5.DataField = "To"
Text6.DataField = "From"
Text7.DataField = "CC's"
Text8.DataField = "Description"
Text9.DataField = "Comments"
Text10.DataField = "Tag 1"
Text11.DataField = "Tag 2"
Text12.DataField = "Tag 3"
Text13.DataField = "Tag 4"
Text14.DataField = "Tag 5"
Text15.DataField = "Tag 6"
Text16.DataField = "Tag 7"
Text17.DataField = "Tag 8"
Text18.DataField = "Tag 9"
```

```
Form1 - 3

Text19.DataField = "Tag 10"
Text20.DataField = "Tag 11"
Text21.DataField = "Tag 12"
Text22.DataField = "Tag 13"
Text23.DataField = "Tag 14"
Text24.DataField = "Tag 15"
Text25.DataField = "Tag 16"
Text26.DataField = "Tag 17"
Text27.DataField = "Tag 18"
Text28.DataField = "Tag 19"
Text29.DataField = "Tag 20"
Text30.DataField = "Tag 21"
Text31.DataField = "Tag 22"
Text32.DataField = "Tag 23"
Text33.DataField = "Tag 24"
Text34.DataField = "Entry Info"
Text35.DataField = "DbPath"
Text35.DataField = "Case Name"
Text36.DataField = "Exhibit #"
Text37.DataField = "Ext"
Label10.DataField = "Field 1"
Label11.DataField = "Field 2"
Label12.DataField = "Field 3"
Label13.DataField = "Field 4"
Label14.DataField = "Field 5"
Label15.DataField = "Field 6"
Label16.DataField = "Field 7"
Label17.DataField = "Field 8"
Label18.DataField = "Field 9"
Label19.DataField = "Field 10"
Label20.DataField = "Field 11"
Label21.DataField = "Field 12"
Label22.DataField = "Field 13"
Label23.DataField = "Field 14"
Label24.DataField = "Field 15"
Label25.DataField = "Field 16"
Label26.DataField = "Field 17"
Label27.DataField = "Field 18"
Label28.DataField = "Field 19"
Label29.DataField = "Field 20"
Label30.DataField = "Field 21"
Label31.DataField = "Field 22"
Label32.DataField = "Field 23"
Label33.DataField = "Field 24"
Label34.DataField = "ImagePath"
ImageMan1.AutoScale = 1
ImageMan1.ScaleMethod = 3
Label35.DataField = "DbPath"


' normal cursor
MousePointer = 0
End Sub
Private Sub Form_Paint()
Data1.Refresh
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
End
End Sub

Private Sub Fulltext_Click()
Rem Remove hard coding for Isys
Dim ISYS$, X&
Rem X%
Label34.DataField = "IsysPath"
ISYS = Label34.Caption
ISYS = "C:\ISYS\IQW.EXE /Z /D=" + ISYS
Label34.DataField = "ImagePath"
```

```
Form1 - 4

X = Shell(ISYS, 1)
End Sub
Private Sub ImageMan1 DblClick()
Rem remove hard coding for watermark
Dim Temp As String, R As Double
Label34.DataField = "ImagePath"
Temp = Label34.Caption & Text34.Text
R = Shell("C:\WMPRO\WMPRO.EXE " & Temp, 3)
End Sub
Private Sub ImageMan1 GotFocus()
Data1.Recordset.MoveNext: Data1.Recordset.MovePrevious
End Sub
Private Sub Text1 DblClick()
gs FieldName$ = "[Doc Number]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text10 DblClick()
gs FieldName$ = "[Tag 1]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text11 DblClick()
gs FieldName$ = "[Tag 2]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text12 DblClick()
gs FieldName$ = "[Tag 3]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text13 DblClick()
gs FieldName$ = "[Tag 4]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text14 DblClick()
gs FieldName$ = "[Tag 5]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text15 DblClick()
gs FieldName$ = "[Tag 6]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text16 DblClick()
gs FieldName$ = "[Tag 7]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text17 DblClick()
gs FieldName$ = "[Tag 8]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text18 DblClick()
gs FieldName$ = "[Tag 9]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text19 DblClick()
gs FieldName$ = "[Tag 10]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text2 DblClick()
gs_FieldName$ = "[Doc Date]"
```

```
Form1 - 5

Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text20 DblClick()
qs FieldName$ = "[Tag 11]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text21 DblClick()
qs FieldName$ = "[Tag 12]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text22 DblClick()
qs FieldName$ = "[Tag 13]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text23 DblClick()
qs FieldName$ = "[Tag 14]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text24 DblClick()
qs FieldName$ = "[Tag 15]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text25 DblClick()
qs FieldName$ = "[Tag 16]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text26 DblClick()
qs FieldName$ = "[Tag 17]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text27 DblClick()
qs FieldName$ = "[Tag 18]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text28 DblClick()
qs FieldName$ = "[Tag 19]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text29 DblClick()
qs FieldName$ = "[Tag 20]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text3 DblClick()
qs FieldName$ = "[Beg Bates #]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text30 DblClick()
qs FieldName$ = "[Tag 21]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text31 DblClick()
qs FieldName$ = "[Tag 22]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text32_DblClick()
```

```
Form1 - 6

qs FieldName$ = "[Tag 23]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text33 DblClick()
qs FieldName$ = "[Tag 24]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text34 Change()
Dim MyDb As String, MyDrive As String, Z As Integer, FileLoc As String
Dim Test As String, BegDocNum As Variant, EndDocNum As Variant, MyPath As String
Dim DocToFind As Variant, CdPresent As String, MyName As String
Dim RemovePrefix As String, X As Double, Msg As String, OldPath As String, Inser
tDisk As Integer
Label36.Caption = ""

Z = Len(Label35.Caption) - 4: MyDb = Left(Label35.Caption, Z)
MyDrive = Left(MyDb, 3)

On Error GoTo ErrorHandler
LoadDriveType

If Text34.Text = "" Then Label36.Caption = "File not Found.  Check Disk"
ImageMan1.Picture = Label34.Caption & Text34.Text
If Text34.Text = "Q.TIF" Then
    ImageMan1.Picture = Label34.Caption & "Q.TIF"
    Label36.Caption = "Picture file not available."
    Exit Sub
    End If
If DriveType = "CD-ROM Drive" Then
    X = Len(Label35.Caption): FileLoc = Left(Label35.Caption, X - 4)
    Open FileLoc & "\CDINFO.DAT" For Input As 1#
    DocToFind = Text34.Text
    Input #1, CDROM, RemovePrefix
    MyName = Mid(DocToFind, RemovePrefix): X = Len(DocToFind)
    MyName = Left(MyName, X - 5): DocToFind = Val(MyName)

    Do While Not EOF(1)
    Input #1, DiskNum, BegDocNum, EndDocNum
    If DocToFind >= BegDocNum And DocToFind <= EndDocNum Then Exit Do
    Loop
    Close #1
    If InsertDisk = 0 Then
        Me.Caption = "Paper Chaser - Disk #" & DiskNum & " in CD ROM."
        Else
        Me.Caption = "Paper Chaser - Insert disk #" & DiskNum & " in CD ROM."
        End If
    End If

If DriveType = "Network Drive" Or DriveType = "Hard Drive" Then
    Me.Caption = "Paper Chaser - " & Label34.Caption & Text34.Text
    End If
Rem Me.Caption = "Paper Chaser - Document Info    " & Label34.Caption & Text34.Te
xt
Exit Sub

ErrorHandler:
Select Case Err.Number
        Case 53 '"File Not Found"
        Label36.Caption = "File not found."
        ImageMan1.Picture = MyDrive & "PChaser\" & "Q.TIF"
        InsertDisk = 1
        Resume Next
        Case 76 ' "No Disk" error.
        Label36.Caption = "No Disk in CDROM. Insert a Disk."
        InsertDisk = 1
        Resume Next
        Case 68 ' "Device not available"
```

```
Form1 - 7

        Label36.Caption = "No Disk in CDROM. Insert a Disk."
        ImageMan1.Picture = MyDrive & "PChaser\" & "Q.TIF"
        InsertDisk = 1
        On Error Resume Next
        Case 32504 '"File Not Found"
        Label36.Caption = "File not found."
        ImageMan1.Picture = MyDrive & "PChaser\" & "Q.TIF"
        InsertDisk = 1
        Resume Next
        Case Else
        Rem On Error Resume Next
        MsgBox Err.Number & "Load Error Number"
        Resume Next
    End Select
    Resume
End Sub

Private Sub Text34_DblClick()
gs FieldName$ = "[Entry Info]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text36_DblClick()
gs FieldName$ = "[Exhibit #]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text37_DblClick()
gs FieldName$ = "[Ext]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text4_DblClick()
gs FieldName$ = "[End Bates #]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text5_DblClick()
gs FieldName$ = "[to]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text6_DblClick()
gs FieldName$ = "[from]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text7_DblClick()
gs FieldName$ = "[CC's]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text8_DblClick()
gs FieldName$ = "[Description]"
Load FrmSearch
FrmSearch.Show
End Sub
Private Sub Text9_DblClick()
gs FieldName$ = "[Comments]"
Load FrmSearch
FrmSearch.Show
End Sub
Function FileExists(Filename As String) As Boolean
Dim TempAttr As Integer
If (Len(Filename) = 0) Or (InStr(Filename, "*") > 0) Or (InStr(Filename, "?") >
0) Then
    FileExists = False
    Exit Function
    End If
```

Form1 - 8

```
On Error GoTo ErrorFileExist
TempAttr = GetAttr(Filename)
FileExists = ((TempAttr And vbDirectory) = 0)
GoTo ExitFileExist

ErrorFileExist:
FileExists = False
Resume ExitFileExist

ExitFileExist:
On Error GoTo 0
End Function
Private Sub LoadDriveType()
Dim MyDrive As String
On Error GoTo ErrorHandler
' get drive type of currently selected drive
MyDrive = Left(Label34.Caption, 1): Drive1.Drive = MyDrive
Select Case GetDriveType(UCase(Left(Drive1.Drive, 1)) & ":\")
      Case DRIVE REMOVABLE
         DriveType = "Floppy Drive"
      Case DRIVE FIXED
         DriveType = "Hard Drive"
      Case DRIVE REMOTE
         DriveType = "Network Drive"
      Case DRIVE CDROM
         DriveType = "CD-ROM Drive"
      Case DRIVE RAMDISK
         DriveType = "RAM Disk"
      Case 0
         DriveType = "Could not determine drive type."
      Case 1
         DriveType = "Drive does not exist."
      Case Else
         DriveType = "Unknown or Illegal Drive"
   End Select
   'MsgBox DriveType
Exit Sub

ErrorHandler:
Select Case Err.Number
    Case 68 'No Drive Found - Occurs when no CD ROM is in Drive
        Me.Caption = "Paper Chaser - No Disk in Drive."
    Case Else
        Rem MsgBox Err.Number & "Drive Type Error Number"
    End Select
End Sub
Private Sub Drive1 Change()
'load the drive type label when the drive combo
' changes
LoadDriveType
End Sub
```

```
FrmRpt - 1

Option Explicit
Private Sub BitRpt_Click()
Dim LoadAccess As String, X As Variant
ProgramPath.DataField = "DbPath"
LoadAccess = "msarn200.exe " & ProgramPath.Caption & " /ini pchaser.ini"
    X = Shell(LoadAccess, 1)
End Sub
Private Sub CmdLoad_Click()
On Error Resume Next
Dim Tempcase$, ReportFileName$
Tempcase = UCase(List1.Text)
Select Case Tempcase
      Case UCase(Form1.Label10.Caption)
         ReportFileName$ = "RPT01.RPT"
      Case UCase(Form1.Label11.Caption)
         ReportFileName$ = "RPT02.RPT"
      Case UCase(Form1.Label12.Caption)
         ReportFileName$ = "RPT03.RPT"
      Case UCase(Form1.Label13.Caption)
         ReportFileName$ = "RPT04.RPT"
      Case UCase(Form1.Label14.Caption)
         ReportFileName$ = "RPT05.RPT"
      Case UCase(Form1.Label15.Caption)
         ReportFileName$ = "RPT06.RPT"
      Case UCase(Form1.Label16.Caption)
         ReportFileName$ = "RPT07.RPT"
      Case UCase(Form1.Label17.Caption)
         ReportFileName$ = "RPT08.RPT"
      Case UCase(Form1.Label18.Caption)
         ReportFileName$ = "RPT09.RPT"
      Case UCase(Form1.Label19.Caption)
         ReportFileName$ = "RPT10.RPT"
      Case UCase(Form1.Label20.Caption)
         ReportFileName$ = "RPT11.RPT"
      Case UCase(Form1.Label21.Caption)
         ReportFileName$ = "RPT12.RPT"
      Case UCase(Form1.Label22.Caption)
         ReportFileName$ = "RPT13.RPT"
      Case UCase(Form1.Label23.Caption)
         ReportFileName$ = "RPT14.RPT"
      Case UCase(Form1.Label24.Caption)
         ReportFileName$ = "RPT15.RPT"
      Case UCase(Form1.Label25.Caption)
         ReportFileName$ = "RPT16.RPT"
      Case UCase(Form1.Label26.Caption)
         ReportFileName$ = "RPT17.RPT"
      Case UCase(Form1.Label27.Caption)
         ReportFileName$ = "RPT18.RPT"
      Case UCase(Form1.Label28.Caption)
         ReportFileName$ = "RPT19.RPT"
      Case UCase(Form1.Label29.Caption)
         ReportFileName$ = "RPT20.RPT"
      Case UCase(Form1.Label30.Caption)
         ReportFileName$ = "RPT21.RPT"
      Case UCase(Form1.Label31.Caption)
         ReportFileName$ = "RPT22.RPT"
      Case UCase(Form1.Label32.Caption)
         ReportFileName$ = "RPT23.RPT"
      Case UCase(Form1.Label33.Caption)
         ReportFileName$ = "RPT24.RPT"
      Case Else
         ReportFileName$ = List1.Text & ".rpt"
   End Select

CrystalReport1.DataFiles(0) = App.Path & "\" & qs_FileName$
CrystalReport1.ReportFileName = App.Path & "\" & ReportFileName

On Error GoTo ErrorHandler
    FrmRpt.Hide
```

FrmRpt - 2

```
    Form1.Refresh
    CrystalReport1.Action = 1
    Exit Sub
ErrorHandler:
    MsgBox CrystalReport1.LastErrorString
Exit Sub
End Sub
Private Sub Command2_Click()
Unload FrmRpt
End Sub
Public Sub GetReportNames()
 List1.Clear
   Dim I As Integer, X As Integer, ReportFileName As String, ReportName As Strin
g
   Dim currentCharacter As String
   File1.Path = App.Path
   For I = 0 To File1.ListCount - 1
      ReportFileName = File1.List(I)
      ' The following code looks at the files in File1
      ' and strips out ".rpt" from it
      For X = 1 To 64
         currentCharacter$ = Mid$(ReportFileName, X, 1)
         If currentCharacter$ = "." Then Exit For
         ReportName$ = ReportName$ + currentCharacter$
      Next X

      ' the following hunk of code should looks at the report name
      ' if it's on of the 24 standard reports, it uses the name
      ' assigned it by the attorney, other wise it uses the name
      ' as it appears in the directory
      Select Case UCase(ReportName$)
         Case UCase("RPT01")
        List1.AddItem Form1.Label10.Caption
         Case UCase("RPT02")
        List1.AddItem Form1.Label11.Caption
         Case UCase("RPT03")
        List1.AddItem Form1.Label12.Caption
         Case UCase("RPT04")
        List1.AddItem Form1.Label13.Caption
         Case UCase("RPT05")
        List1.AddItem Form1.Label14.Caption
         Case UCase("RPT06")
        List1.AddItem Form1.Label15.Caption
         Case UCase("RPT07")
        List1.AddItem Form1.Label16.Caption
         Case UCase("RPT08")
        List1.AddItem Form1.Label17.Caption
         Case UCase("RPT09")
        List1.AddItem Form1.Label18.Caption
         Case UCase("RPT10")
        List1.AddItem Form1.Label19.Caption
         Case UCase("RPT11")
        List1.AddItem Form1.Label20.Caption
         Case UCase("RPT12")
        List1.AddItem Form1.Label21.Caption
         Case UCase("RPT13")
        List1.AddItem Form1.Label22.Caption
         Case UCase("RPT14")
        List1.AddItem Form1.Label23.Caption
         Case UCase("RPT15")
        List1.AddItem Form1.Label24.Caption
         Case UCase("RPT16")
        List1.AddItem Form1.Label25.Caption
         Case UCase("RPT17")
        List1.AddItem Form1.Label26.Caption
         Case UCase("RPT18")
        List1.AddItem Form1.Label27.Caption
         Case UCase("RPT19")
        List1.AddItem Form1.Label28.Caption
```

```
FrmRpt - 3

         Case UCase("RPT20")
        List1.AddItem Form1.Label29.Caption
         Case UCase("RPT21")
        List1.AddItem Form1.Label30.Caption
         Case UCase("RPT22")
        List1.AddItem Form1.Label31.Caption
         Case UCase("RPT23")
        List1.AddItem Form1.Label32.Caption
         Case UCase("RPT24")
        List1.AddItem Form1.Label33.Caption
         Case Else
        List1.AddItem ReportName$
        End Select
        ReportName$ = ""         'reset reportName$ to nothing
   Next I
End Sub
Private Sub Form Load()
On Error GoTo ErrorHandler
Me.Left = (Screen.Width - Me.Width) / 2: Me.Top = (Screen.Height - Me.Height) /
2
Data1.DatabaseName = App.Path & "\" & gs_FileName$
ProgramPath.DataField = "DbPath"
Data1.RecordSource = "Select * from [Preferences]"
Call GetReportNames
Exit Sub
ErrorHandler:
Select Case Err.Number
        Case 0
            Resume Next
        Case Else
            Rem On Error Resume Next
            MsgBox Err.Number & "Load Error Number"
            Resume Next
        End Select
End Sub
Private Sub List1 DblClick()
CmdLoad.Value = True
End Sub
```

FrmSearch - 1

```
Option Explicit
Private Sub CmdCancel_Click()
Unload FrmSearch
End Sub

Private Sub CmdFirst_Click()
Dim MyCriteria As String
Form1.Data1.Recordset.MoveFirst
Form1.Data2.Recordset.MoveFirst
MyCriteria = gs FieldName$ & " like " & "'*" & txtSearch.Text & "*'"
Form1.Data1.Recordset.FindFirst MyCriteria
If Form1.Data1.Recordset.NoMatch Then
    MsgBox "Term not found in search."
    End If
End Sub

Private Sub CmdNext_Click()
Dim MyCriteria As String
MyCriteria = gs FieldName$ & " like " & "'*" & txtSearch.Text & "*'"
Form1.Data1.Recordset.FindNext MyCriteria
If Form1.Data1.Recordset.NoMatch Then
    MsgBox "Term not found in search."
    End If
End Sub

Private Sub Form_Load()
Me.Left = (Screen.Width - Me.Width) / 2
Me.Top = (Screen.Height - Me.Height) / 2
End Sub
```

```
GotoDoc - 1

Option Explicit
Private Sub CmdCancel_Click()
Unload GotoDoc
End Sub

Private Sub CmdFirst Click()
Dim MyCriteria As String
Rem Form1.Data1.Recordset.MoveFirst
MyCriteria = qs FieldName$ & " like " & "'*" & txtSearch.Text & "*'"
Form1.Data1.Recordset.FindFirst MyCriteria
If Form1.Data1.Recordset.NoMatch Then
    MsgBox "Term not found in search."
    End If
End Sub

Private Sub Form Load()
Me.Left = (Screen.Width - Me.Width) / 2
Me.Top = (Screen.Height - Me.Height) / 2
End Sub
```

```
Module1 - 1

Public qs FieldName$
Public qs FileName$
Public CurrentDisk As Variant
Public DiskNum As Variant
Public MyCriteria As String
```

frmImageViewer - 1

```
'Document Image Viewer - Doc I.V.
'Copyright (c) 1996, Tempest Software Inc.
Option Explicit
DefInt A-Z
Private Declare Function WritePrivateProfileString Lib "Kernel" (ByVal lpApplica
tionName As String, lpKeyName As Any, lpString As Any, ByVal lplFileName As Stri
ng) As Integer
Dim sImgFileName As String
Dim iNumOfPages As Integer
Dim iCurrentPage As Integer
Dim iQualityFactor As Integer
'General Constants
Const SCROLL_BAR = 250
Const PROGRAM_NAME = "Doc I.V."
'VisualBasic Constants
Const OFN_HIDEREADONLY = &H4&
Const PD_PRINTSETUP = &H40&
'TMS Constants
Const VX_FULLIMAGE = 0
Const VX_ZOOMIN = 1
Const VX_ZOOMRECT = 3
Const VX_STARTPRINT = 5
Const VX_PRINT = 6
Const VX_ENDPRINT = 7
Const VX_MARKHOLLOW = 2
Const VX_MAGNIFIER = 3
Const VX_NONE = 0
Const VX_HORIZONTAL = 1
Const VX_VERTICAL = 2
Const VX_BOTH = 3
Const VX_BESTFIT = 1
Const VX_FULLWIDTH = 2
Const VX_FULLHEIGHT = 3
'TMS Error Constants
Const ERR_BADFILENAME = 20000
Const ERR_BADPROPVALUE = 20001
Const ERR_NOIMAGE = 20002
Const ERR_NOPRINTER = 20003
Const ERR_GETPALETTE = 20004
Const ERR_INTERROR = 20005
Const ERR_BADDC = 20006
Const ERR_BAD_WINDOW = 20500
Const ERR_CLIPBOARD_ERROR = 20501
Const ERR_BITMAP_ERROR = 20502
Const ERR_NO_PRINTER = 20503
Const ERR_NO_GRAPHICS = 20504
Const ERR_PRINT_ABORTED = 20505
Const ERR_BAD_PAGENUM = 20506
Const ERR_PRINT_BUSY = 20507
Const ERR_BAD_PERCENT = 20508
Const ERR_NO_RECT = 20509
Const ERR_CANT_ZOOM = 20510
Const ERR_CANT_SCROLL = 20511
Const ERR_NO_MEMORY = 20512
Const ERR_BAD_PARM = 20513

Private Sub exitProgram()
    Call preferencesSave
    Unload frmImageViewer
End Sub

Private Sub fileOpen()
    On Error Resume Next
    frmImageViewer.vdVBX.Visible = False
    frmImageViewer.vdVBX.filename = sImgFileName$
    iNumOfPages% = vdVBX.Pages
    iCurrentPage% = vdVBX.Page + 1
    HScroll1.Value = 0
```

frmImageViewer - 2

```
   Select Case iNumOfPages%
   Case 0
      HScroll1.Visible = False
      MsgBox "The specified file name is" & Chr(10) & "either not a valid file n
ame," & Chr(10) & "or is not in an accepted image format.", 0, "Error loading fi
le."
      frmImageViewer.Caption = PROGRAM_NAME
      Exit Sub
   Case 1
      HScroll1.Visible = False
      frmImageViewer.Caption = frmImageViewer.CMDialog1.FileTitle
   Case Else
      HScroll1.Visible = True
      frmImageViewer.Caption = frmImageViewer.CMDialog1.FileTitle & "    Page " &
 iCurrentPage% & " of " & iNumOfPages%
   End Select
   Call menuEnabled
   Call imageRefresh
End Sub

Private Sub Form_Load()
   Call preferencesLoad
   Call menuDisabled
   frmImageViewer.Caption = PROGRAM_NAME
   frmImageViewer.vdVBX.Visible = False
   frmImageViewer.vdVBX.MagnifyRatio = 2
   frmImageViewer.vdVBX.ZoomRatio = 50
   vdVBX.RightMouseStyle = VX_MAGNIFIER
   vdVBX.LeftMouseStyle = VX_MARKHOLLOW
   HScroll1.Visible = False
   If gFileName$ <> "" Then
      sImgFileName$ = gFileName$
      Call fileOpen
   End If
End Sub

Private Sub Form_Resize()
   On Error Resume Next
   frmImageViewer.vdVBX.Visible = False
   vdVBX.Width = frmImageViewer.ScaleWidth
   vdVBX.Height = frmImageViewer.ScaleHeight - vdVBX.Top
   HScroll1.Left = 20
   HScroll1.Top = frmImageViewer.ScaleHeight - HScroll1.Height - 217
   Call imageRefresh
End Sub

Private Sub Form_Unload(Cancel As Integer)
   Call exitProgram
End Sub

Private Sub HScroll1_Change()
   On Error Resume Next
   HScroll1.Max = iNumOfPages - 1
   HScroll1.Min = 0
   frmImageViewer.vdVBX.Visible = False
   frmImageViewer.vdVBX.Page = HScroll1.Value
   iCurrentPage = HScroll1.Value + 1
   frmImageViewer.Caption = frmImageViewer.CMDialog1.FileTitle & "    Page " & iC
urrentPage% & " of " & iNumOfPages%
   Call imageRefresh
End Sub

Private Sub imageRefresh()
   On Error Resume Next
   If mnuPreferencesFitHeight.Checked = True Then
      frmImageViewer.vdVBX.ImageScaleHeight = vdVBX.Height - SCROLL_BAR
   ElseIf mnuPreferencesFitWidth.Checked = True Then
      frmImageViewer.vdVBX.ImageScaleWidth = vdVBX.Width - SCROLL_BAR
   End If
```

```
frmImageViewer - 3

   frmImageViewer.vdVBX.ImageScaleLeft = 0
   frmImageViewer.vdVBX.ImageScaleTop = 0
   frmImageViewer.vdVBX.Visible = True
   Call menuPageCheck
   frmImageViewer.vdVBX.SetFocus
End Sub

Private Sub menuDisabled()
   mnuFilePrint.Enabled = False
   mnuPageNext.Enabled = False
   mnuPagePrevious.Enabled = False
   mnuPageFirst.Enabled = False
   mnuPageLast.Enabled = False
   mnuPageGoto.Enabled = False
   mnuZoomZoomIn.Enabled = False
   mnuZoomZoomOut.Enabled = False
   mnuEffectsRotateC.Enabled = False
   mnuEffectsRotateCC.Enabled = False
   mnuEffectsRotateImageFlip.Enabled = False
   mnuEffectsRotateReset.Enabled = False
   mnuEffectsMirrorH.Enabled = False
   mnuEffectsMirrorV.Enabled = False
   mnuEffectsMirrorBoth.Enabled = False
   mnuEffectsMirrorReset.Enabled = False
   mnuPreferencesInvertColors.Enabled = False
End Sub

Private Sub menuEnabled()
   mnuFilePrint.Enabled = True
   mnuPageNext.Enabled = True
   mnuPagePrevious.Enabled = True
   mnuPageFirst.Enabled = True
   mnuPageLast.Enabled = True
   mnuPageGoto.Enabled = True
   mnuZoomZoomIn.Enabled = True
   mnuZoomZoomOut.Enabled = True
   mnuEffectsRotateC.Enabled = True
   mnuEffectsRotateCC.Enabled = True
   mnuEffectsRotateImageFlip.Enabled = True
   mnuEffectsRotateReset.Enabled = True
   mnuEffectsMirrorH.Enabled = True
   mnuEffectsMirrorV.Enabled = True
   mnuEffectsMirrorBoth.Enabled = True
   mnuEffectsMirrorReset.Enabled = True
   mnuPreferencesInvertColors.Enabled = True
End Sub

Private Sub menuPageCheck()
   If vdVBX.Pages = 0 Then Exit Sub
' check for page numbers if only one page disable all pageroutines
   If vdVBX.Pages = 1 Then
      mnuPageNext.Enabled = False
      mnuPagePrevious.Enabled = False
      mnuPageFirst.Enabled = False
      mnuPageLast.Enabled = False
      mnuPageGoto.Enabled = False
      Exit Sub
   End If
'check for 1st page
   If HScroll1.Value = 0 Then
      mnuPagePrevious.Enabled = False
      mnuPageFirst.Enabled = False
      mnuPageNext.Enabled = True
      mnuPageLast.Enabled = True
   ElseIf HScroll1.Value = (iNumOfPages - 1) Then
      mnuPagePrevious.Enabled = True
      mnuPageFirst.Enabled = True
      mnuPageNext.Enabled = False
      mnuPageLast.Enabled = False
```

```
frmImageViewer - 4

   Else
      mnuPagePrevious.Enabled = True
      mnuPageFirst.Enabled = True
      mnuPageNext.Enabled = True
      mnuPageLast.Enabled = True
End If
End Sub

Private Sub mnuEffectsMirrorBoth_Click()
   Select Case frmImageViewer.vdVBX.Mirror
   Case VX_NONE
      frmImageViewer.vdVBX.Mirror = VX_BOTH
   Case VX_BOTH
      frmImageViewer.vdVBX.Mirror = VX_NONE
   Case VX_VERTICAL
      frmImageViewer.vdVBX.Mirror = VX_HORIZONTAL
   Case VX_HORIZONTAL
      frmImageViewer.vdVBX.Mirror = VX_VERTICAL
   End Select
   imageRefresh
End Sub

Private Sub mnuEffectsMirrorH_Click()
   Select Case frmImageViewer.vdVBX.Mirror
   Case VX_NONE
      frmImageViewer.vdVBX.Mirror = VX_HORIZONTAL
   Case VX_HORIZONTAL
      frmImageViewer.vdVBX.Mirror = VX_NONE
   Case VX_BOTH
      frmImageViewer.vdVBX.Mirror = VX_VERTICAL
   Case VX_VERTICAL
      frmImageViewer.vdVBX.Mirror = VX_BOTH
   End Select
   imageRefresh
End Sub

Private Sub mnuEffectsMirrorReset_Click()
   frmImageViewer.vdVBX.Mirror = VX_NONE
   imageRefresh
End Sub

Private Sub mnuEffectsMirrorV_Click()
   Select Case frmImageViewer.vdVBX.Mirror
   Case VX_NONE
      frmImageViewer.vdVBX.Mirror = VX_VERTICAL
   Case VX_VERTICAL
      frmImageViewer.vdVBX.Mirror = VX_NONE
   Case VX_BOTH
      frmImageViewer.vdVBX.Mirror = VX_HORIZONTAL
   Case VX_HORIZONTAL
      frmImageViewer.vdVBX.Mirror = VX_BOTH
   End Select
   imageRefresh
End Sub

Private Sub mnuEffectsRotateC_Click()
   Select Case frmImageViewer.vdVBX.Rotation
   Case 0
      frmImageViewer.vdVBX.Rotation = 90
   Case 90
      frmImageViewer.vdVBX.Rotation = 180
   Case 180
      frmImageViewer.vdVBX.Rotation = 270
   Case 270
      frmImageViewer.vdVBX.Rotation = 0
   End Select
   imageRefresh
End Sub
```

frmImageViewer - 5

```
Private Sub mnuEffectsRotateCC_Click()
   Select Case frmImageViewer.vdVBX.Rotation
   Case 0
      frmImageViewer.vdVBX.Rotation = 270
   Case 90
      frmImageViewer.vdVBX.Rotation = 0
   Case 180
      frmImageViewer.vdVBX.Rotation = 90
   Case 270
      frmImageViewer.vdVBX.Rotation = 180
   End Select
   imageRefresh
End Sub

Private Sub mnuEffectsRotateImageFlip_Click()
   Select Case frmImageViewer.vdVBX.Rotation
   Case 0
      frmImageViewer.vdVBX.Rotation = 180
   Case 90
      frmImageViewer.vdVBX.Rotation = 270
   Case 180
      frmImageViewer.vdVBX.Rotation = 0
   Case 270
      frmImageViewer.vdVBX.Rotation = 90
   End Select
   imageRefresh
End Sub

Private Sub mnuEffectsRotateReset_Click()
   frmImageViewer.vdVBX.Rotation = 0
   imageRefresh
End Sub

Private Sub mnuFileExit_Click()
   Call exitProgram
End Sub

Private Sub mnuFileOpen_Click()
   On Error GoTo loadError
   frmImageViewer.vdVBX.Visible = False
   frmImageViewer.HScroll1.Visible = False
   frmImageViewer.Caption = PROGRAM_NAME
   frmImageViewer.CMDialog1.CancelError = True
   frmImageViewer.CMDialog1.DialogTitle = "Open Image File"
   frmImageViewer.CMDialog1.Flags = OFN_HIDEREADONLY
   frmImageViewer.CMDialog1.Filter = "All Images (*.*)|*.bmp;*.jpg;*.pcx;*.tif;|
JPEG files (*.jpg)| *.jpg|PCX files (*.pcx) |*.pcx |TIFF files (*.tif) | *.tif |
Windows Bitmap (*.bmp) |*.bmp "
   frmImageViewer.CMDialog1.FilterIndex = 1
   frmImageViewer.CMDialog1.Action = 1
   sImgFileName$ = frmImageViewer.CMDialog1.filename
   frmImageViewer.vdVBX.Quality = iQualityFactor%
   Call fileOpen
   Exit Sub
loadError:
   frmImageViewer.Caption = PROGRAM_NAME
   Exit Sub
End Sub

Private Sub mnuFilePrint_Click()
   vdVBX.PrintStyle = VX_FULLIMAGE
   vdVBX.Action = VX_STARTPRINT
   vdVBX.Action = VX_PRINT
   vdVBX.Action = VX_ENDPRINT
End Sub

Private Sub mnuFilePrintSetup_Click()
   Dim CancelFlag As Integer
   CancelFlag = True
```

```
frmImageViewer - 6

   On Error Resume Next
   CMDialog1.CancelError = True
   CMDialog1.Flags = PD_PRINTSETUP
   CMDialog1.Action = 5
   If (Err = 0) Then
      CancelFlag = False
   End If
   If (CancelFlag = True) Then Exit Sub
End Sub

Private Sub mnuHelpAbout_Click()
   Load frmAbout
   frmAbout.Show 1
   frmImageViewer.Show
End Sub

Private Sub mnuPageFirst_Click()
   HScroll1.Value = 0
End Sub

Private Sub mnuPageGoto_Click()
   Load frmGotoPage
   frmGotoPage.Show 1
End Sub

Private Sub mnuPageLast_Click()
   HScroll1.Value = vdVBX.Pages - 1
End Sub

Private Sub mnuPageNext_Click()
   HScroll1.Value = HScroll1.Value + 1
End Sub

Private Sub mnuPagePrevious_Click()
   HScroll1.Value = HScroll1.Value - 1
End Sub

Private Sub mnuPreferencesFitHeight_Click()
   mnuPreferencesFitHeight.Checked = True
   mnuPreferencesFitWidth.Checked = False
   Call imageRefresh
End Sub

Private Sub mnuPreferencesFitWidth_Click()
   mnuPreferencesFitHeight.Checked = False
   mnuPreferencesFitWidth.Checked = True
   Call imageRefresh
End Sub

Private Sub mnuPreferencesImageQualityHigh_Click()
   iQualityFactor% = 10
   mnuPreferencesImageQualityLow.Checked = False
   mnuPreferencesImageQualityMedium.Checked = False
   mnuPreferencesImageQualityHigh.Checked = True
   vdVBX.Quality = iQualityFactor%
End Sub

Private Sub mnuPreferencesImageQualityLow_Click()
   iQualityFactor% = 0
   mnuPreferencesImageQualityLow.Checked = True
   mnuPreferencesImageQualityMedium.Checked = False
   mnuPreferencesImageQualityHigh.Checked = False
   vdVBX.Quality = iQualityFactor%
End Sub

Private Sub mnuPreferencesImageQualityMedium_Click()
   iQualityFactor% = 5
   mnuPreferencesImageQualityLow.Checked = False
   mnuPreferencesImageQualityMedium.Checked = True
```

```
frmImageViewer - 7

   mnuPreferencesImageQualityHigh.Checked = False
   vdVBX.Quality = iQualityFactor%
End Sub

Private Sub mnuPreferencesInvertColors_Click()
   If frmImageViewer.vdVBX.Invert = True Then
      frmImageViewer.vdVBX.Invert = False
      frmImageViewer.mnuPreferencesInvertColors.Checked = False
   Else
      frmImageViewer.vdVBX.Invert = True
      frmImageViewer.mnuPreferencesInvertColors.Checked = True
   End If
End Sub

Private Sub mnuZoomZoomIn_Click()
   Call zoomIn
End Sub

Private Sub mnuZoomZoomOut_Click()
   Call zoomOut
End Sub

Private Sub preferencesLoad()
   frmImageViewer.Left = gFormLeft&
   frmImageViewer.Top = gFormTop&
   frmImageViewer.Width = gFormWidth&
   frmImageViewer.Height = gFormHeight&
   If InStr(gFitWidth$, "TRUE") Then
      mnuPreferencesFitHeight.Checked = False
      mnuPreferencesFitWidth.Checked = True
   Else
      mnuPreferencesFitHeight.Checked = True
      mnuPreferencesFitWidth.Checked = False
   End If
   Select Case gQualityFactor%
   Case 0
      mnuPreferencesImageQualityLow.Checked = True
      mnuPreferencesImageQualityMedium.Checked = False
      mnuPreferencesImageQualityHigh.Checked = False
      iQualityFactor% = 0
   Case 5
      mnuPreferencesImageQualityLow.Checked = False
      mnuPreferencesImageQualityMedium.Checked = True
      mnuPreferencesImageQualityHigh.Checked = False
      iQualityFactor% = 5
   Case 10
      mnuPreferencesImageQualityLow.Checked = False
      mnuPreferencesImageQualityMedium.Checked = False
      mnuPreferencesImageQualityHigh.Checked = True
      iQualityFactor% = 10
   End Select
End Sub

Private Sub preferencesSave()
   'On Error Resume Next
   Dim iRet%
   Dim sSection$
   Dim sEntry$
   Dim sValue$
   Dim sFileName$
   If Right(App.Path, 1) = "\" Then
      sFileName$ = App.Path & "dociv.ini"
   Else
      sFileName$ = App.Path & "\" & "dociv.ini"
   End If
   sSection$ = "Form Position"
   sEntry$ = "Form Width"
   sValue$ = Str(frmImageViewer.Width)
   iRet = WritePrivateProfileString(ByVal sSection, ByVal sEntry, ByVal sValue,
```

```
frmImageViewer - 8

ByVal sFileName)
    sEntry$ = "Form Height"
    sValue$ = Str(frmImageViewer.Height)
    iRet = WritePrivateProfileString(ByVal sSection, ByVal sEntry, ByVal sValue,
ByVal sFileName)
    sEntry$ = "Form Left"
    sValue$ = Str(frmImageViewer.Left)
    iRet = WritePrivateProfileString(ByVal sSection, ByVal sEntry, ByVal sValue,
ByVal sFileName)
    sEntry$ = "Form Top"
    sValue$ = Str(frmImageViewer.Top)
    iRet = WritePrivateProfileString(ByVal sSection, ByVal sEntry, ByVal sValue,
ByVal sFileName)
    sSection$ = "Preferences"
    sEntry$ = "Fit Width"
    If mnuPreferencesFitWidth.Checked = True Then
       sValue$ = "TRUE"
    Else
       sValue$ = "FALSE"
    End If
    iRet = WritePrivateProfileString(ByVal sSection, ByVal sEntry, ByVal sValue,
ByVal sFileName)
    sEntry$ = "Image Quality"
    sValue$ = Str(iQualityFactor%)
    iRet = WritePrivateProfileString(ByVal sSection, ByVal sEntry, ByVal sValue,
ByVal sFileName)
End Sub

Private Sub vdVBX_Status(Index As Integer, Status As Integer, Value As Long)
    On Error Resume Next
    If frmImageViewer.vdVBX.Marked = True Then
       frmImageViewer.vdVBX.Action = VX_ZOOMRECT
    End If
End Sub

Private Sub zoomIn()
    On Error Resume Next
    frmImageViewer.vdVBX.ZoomRatio = 50
    frmImageViewer.vdVBX.Action = VX_ZOOMIN
End Sub

Private Sub zoomOut()
   On Error Resume Next
    frmImageViewer.vdVBX.ZoomRatio = 50
    frmImageViewer.vdVBX.Action = 2 'VX_ZOOMOUT
End Sub
```

```
frmIntro - 1

Option Explicit
DefInt A-Z
Private Declare Function GetPrivateProfileString Lib "Kernel" (ByVal lpApplicati
onName As String, lpKeyName As Any, ByVal lpDefault As String, ByVal lpReturnedS
tring As String, ByVal nSize As Integer, ByVal lpFileName As String) As Integer

Private Sub displayText()
   FontSize = 100
   frmIntro.CurrentX = 22: frmIntro.CurrentY = 52
   ForeColor = QBColor(8)
   Print "Doc I.V."
   frmIntro.CurrentX = 20: frmIntro.CurrentY = 50
   ForeColor = QBColor(15)
   Print "Doc I.V."
   frmIntro.CurrentX = 21: frmIntro.CurrentY = 51
   ForeColor = QBColor(7)
   Print "Doc I.V."
   '*****************************
   FontSize = 18
   frmIntro.CurrentX = 72: frmIntro.CurrentY = 202
   ForeColor = QBColor(8)
   Print "The Multiple Format Image Viewer"
   frmIntro.CurrentX = 70: frmIntro.CurrentY = 200
   ForeColor = QBColor(15)
   Print "The Multiple Format Image Viewer"
   frmIntro.CurrentX = 71: frmIntro.CurrentY = 201
   ForeColor = QBColor(7)
   Print "The Multiple Format Image Viewer"
   '*****************************
   frmIntro.CurrentX = 17: frmIntro.CurrentY = 302
   ForeColor = QBColor(8)
   Print "Doc IV, Copyright (c) 1996, Tempest Software"
   frmIntro.CurrentX = 15: frmIntro.CurrentY = 300
   ForeColor = QBColor(15)
   Print "Doc IV, Copyright (c) 1996, Tempest Software"
   frmIntro.CurrentX = 16: frmIntro.CurrentY = 301
   ForeColor = QBColor(7)
   Print "Doc IV, Copyright (c) 1996, Tempest Software"
End Sub

Private Sub exitSplashScreen()
   frmImageViewer.Show
   frmIntro.Hide
   Unload frmIntro
End Sub

Private Sub Form_Click()
   exitSplashScreen
End Sub

Private Sub Form_KeyPress(KeyAscii As Integer)
   exitSplashScreen
End Sub

Private Sub Form_Load()
   On Error Resume Next
   frmIntro.Left = (Screen.Width - frmIntro.Width) / 2
   frmIntro.Top = (Screen.Height - frmIntro.Height) / 2
   Call preferencesCheck
   gFileName$ = Command$
   Load frmImageViewer
   If gFileName$ <> "" Then
      Load frmImageViewer
      Call exitSplashScreen
   End If
End Sub

Private Sub Form_Paint()
   Call displayText
```

```
frmIntro - 2

End Sub

Private Sub preferencesCheck()
   Dim sFileName$
   Dim sSection$
   Dim sKeyName$
   Dim sDefault$
   Dim sReturn$
   Dim iReturnSize%
   Dim iReturn%
   If Right(App.Path, 1) = "\" Then
      sFileName$ = App.Path & "dociv.ini"
   Else
      sFileName$ = App.Path & "\" & "dociv.ini"
   End If
   sSection$ = "Form Position"
   sReturn$ = Space$(10)
   iReturnSize% = 10
   sKeyName$ = "Form Height"
   sDefault$ = Str(Screen.Height)
   iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$, sDefault
$, sReturn$, iReturnSize%, ByVal sFileName$)
   gFormHeight& = Val(sReturn$)
   sKeyName$ = "Form Width"
   sDefault$ = Str(Screen.Width)
   iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$, sDefault
$, sReturn$, iReturnSize%, ByVal sFileName$)
   gFormWidth& = Val(sReturn$)
   sKeyName$ = "Form Left"
   sDefault$ = "0"
   iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$, sDefault
$, sReturn$, iReturnSize%, ByVal sFileName$)
   gFormLeft& = Val(sReturn$)
   sKeyName$ = "Form Top"
   sDefault$ = "0"
   iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$, sDefault
$, sReturn$, iReturnSize%, ByVal sFileName$)
   gFormTop& = Val(sReturn$)
   sSection$ = "Preferences"
   sKeyName$ = "Fit Width"
   sDefault$ = "TRUE"
   iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$, sDefault
$, sReturn$, iReturnSize%, ByVal sFileName$)
   gFitWidth$ = sReturn$
   sKeyName$ = "Image Quality"
   sDefault$ = "5"
   iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$, sDefault
$, sReturn$, iReturnSize%, ByVal sFileName$)
   gQualityFactor% = Val(sReturn$)
'    sKeyName$ = "Start Up Screen"
'    sDefault$ = "TRUE"
'    iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$, sDefaul
t$, sReturn$, iReturnSize%, ByVal sFilename$)
'    gStartUpScreen$ = sReturn$
'    If InStr(gStartUpScreen$, "FALSE") Then
'       Load frmImageViewer
'       Call exitSplashScreen
'    End If
End Sub

Private Sub Timer1 Timer()
   exitSplashScreen
End Sub
```

```
frmGotoPage - 1

Option Explicit
DefInt A-Z
Dim iPageNumber As Integer

Private Sub cmdOK_Click()
    If iPageNumber > 0 Then
        frmImageViewer.HScroll1.Value = iPageNumber - 1
    End If
    frmGotoPage.Hide
    Unload frmGotoPage
End Sub

Private Sub Form_Load()
    frmGotoPage.Left = (Screen.Width - frmGotoPage.Width) / 2
    frmGotoPage.Top = (Screen.Height - frmGotoPage.Height) / 2
    lblNumOfPages.Caption = "There are " & frmImageViewer.vdVBX.Pages & " pages i
n this file."
End Sub

Private Sub txtGotoPage_Change()
' this must only be a number and it must be > 0 & < total number of pages
    On Error Resume Next
    iPageNumber = txtGotoPage.Text
End Sub


frmAbout - 1

Option Explicit
DefInt A-Z

Private Sub cmdOK_Click()
    frmAbout.Hide
    Unload frmAbout
End Sub

Private Sub Form_Load()
    frmAbout.Left = (Screen.Width - frmAbout.Width) / 2
    frmAbout.Top = (Screen.Height - frmAbout.Height) / 2
End Sub


GLOBALS - 1

Option Explicit
Global gFormWidth&
Global gFormHeight&
Global gFormLeft&
Global gFormTop&
Global gFitWidth$
Global gQualityFactor%
Global gFileName$
'Global gStartUpScreen$
```

X:\PCHASER\NATIONAL.MDB Tuesday, October 21, 1997
Form: Main Page: 38

Text Align:	General	Top:	3120
Visible:	Yes	Width:	1149

Code

```
 1  Option Compare Database   'Use database order for string comparisons
 2
 3  Sub Button23_Click ()
 4  On Error GoTo Err_Button23_Click
 5
 6      Dim DocName As String
 7      Dim LinkCriteria As String
 8
 9      DocName = "Form1"
10      DoCmd OpenForm DocName, , , LinkCriteria
11
12  Exit_Button23_Click:
13      Exit Sub
14
15  Err_Button23_Click:
16      MsgBox Error$
17      Resume Exit_Button23_Click
18
19  End Sub
20
21  Sub Exhibits_Click ()
22  On Error GoTo Err_Exhibits_Click
23
24      Dim DocName As String
25      Dim LinkCriteria As String
26
27      DocName = "Exhibit List"
28      DoCmd OpenForm DocName, , , LinkCriteria
29
30  Exit_Exhibits_Click:
31      Exit Sub
32
33  Err_Exhibits_Click:
34      MsgBox Error$
35      Resume Exit_Exhibits_Click
36
37  End Sub
38
39  Sub Form_GotFocus ()
40  DoCmd Maximize
41  End Sub
42
43  Sub Form_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As
    Single)
44  Me.width = screen.width
```

```
45  Me.height = screen.height
46  Me.left = (screen.width - Me.width) - 2
47  Me.Top = (screen.height - Me.height) - 2
48  End Sub
49
50  Sub Form_MouseUp (Button As Integer, Shift As Integer, X As Single, Y As Single)
51  Me.width = screen.width
52  Me.height = screen.height
53  Me.left = 0
54  Me.Top = 0
55  End Sub
56
57  Sub Form_Open (Cancel As Integer)
58  ahtAccessSystemItems False, True, True
59  End Sub
60
61  Sub Report_Menu_Click ()
62  On Error GoTo Err_Report_Menu_Click
63
64      Dim DocName As String
65      Dim LinkCriteria As String
66
67      DocName = "Print Menu"
68      DoCmd OpenForm DocName, , , LinkCriteria
69
70  Exit_Report_Menu_Click:
71      Exit Sub
72
73  Err_Report_Menu_Click:
74      MsgBox Error$
75      Resume Exit_Report_Menu_Click
76
77  End Sub
78
79  Sub Reports_Menu_Click ()
80  On Error GoTo Err_Reports_Menu_Click
81
82      Dim DocName As String
83      Dim LinkCriteria As String
84
85      DocName = "Print Menu"
86      DoCmd OpenForm DocName, , , LinkCriteria
87
88  Exit_Reports_Menu_Click:
89      Exit Sub
90
91  Err_Reports_Menu_Click:
92      MsgBox Error$
93      Resume Exit_Reports_Menu_Click
94
95  End Sub
```

```
 96
 97  Sub Text_Search_Click ()
 98  Dim ISYS$, X%
 99  ISYS = DLookup("IsysPath", "Preferences")
100  ISYS = "C:\ISYS\IQW.EXE /Z /D=" + ISYS
101  X = Shell(ISYS, 1)
102  End Sub
103
104  Sub Transcript_Click ()
105  On Error GoTo Err_Transcript_Click
106
107      Dim X As Integer
108      Dim AppName As String
109
110      AppName = "C:\ACCESS\SUMCLONE.EXE"
111      X = Shell(AppName, 1)
112
113  Exit_Transcript_Click:
114      Exit Sub
115
116  Err_Transcript_Click:
117      MsgBox Error$
118      Resume Exit_Transcript_Click
119
120  End Sub
121
122  Sub Transcript_Search_Click ()
123  On Error GoTo Err_Transcript_Search_Click
124
125      Dim X As Integer
126      Dim AppName As String
127
128      AppName = "SUMCLONE.EXE"
129      X = Shell(AppName, 1)
130
131  Exit_Transcript_Search_Click:
132      Exit Sub
133
134  Err_Transcript_Search_Click:
135      MsgBox Error$
136      Resume Exit_Transcript_Search_Click
137
138  End Sub
139
140  Sub Transcripts_Click ()
141  On Error GoTo Err_Transcripts_Click
142
143      Dim X As Integer
144      Dim AppName As String
145
146      AppName = "SUMCLONE.EXE"
```

```
147       X = Shell(AppName, 1)
148
149   Exit_Transcripts_Click:
150       Exit Sub
151
152   Err_Transcripts_Click:
153       MsgBox Error$
154       Resume Exit_Transcripts_Click
155
156   End Sub
157
```

Scroll Bars:	None	Section:	0
Special Effect:	Sunken	Tab Index:	4
Tab Stop:	Yes	Text Align:	General
Top:	1560	Visible:	Yes
Width:	3600		

Code

```
 1  Option Compare Database   'Use database order for string comparisons
 2
 3  Sub Return_to_System_Men_Click ()
 4  On Error GoTo Err_Return_to_System_Men_Click
 5
 6
 7      DoCmd Close
 8
 9  Exit_Return_to_System_Men_Click:
10      Exit Sub
11
12  Err_Return_to_System_Men_Click:
13      MsgBox Error$
14      Resume Exit_Return_to_System_Men_Click
15
16  End Sub
17
```

```
28  Exit_Comments_Click:
29      Exit Sub
30
31  Err_Comments_Click:
32      MsgBox Error$
33      Resume Exit_Comments_Click
34
35  End Sub
36
37  Sub Date_Range_Click ()
38  On Error GoTo Err_Date_Range_Click
39
40      Dim DocName As String
41
42      DocName = "Date Range"
43      DoCmd OpenReport DocName, A_PREVIEW
44
45  Exit_Date_Range_Click:
46      Exit Sub
47
48  Err_Date_Range_Click:
49      MsgBox Error$
50      Resume Exit_Date_Range_Click
51
52  End Sub
53
54  Sub Description_Click ()
55  On Error GoTo Err_Description_Click
56
57      Dim DocName As String
58
59      DocName = "Search Term in Description"
60      DoCmd OpenReport DocName, A_PREVIEW
61
62  Exit_Description_Click:
63      Exit Sub
64
65  Err_Description_Click:
66      MsgBox Error$
67      Resume Exit_Description_Click
68
69  End Sub
70
71  Sub Doc___Find_Click ()
72  On Error GoTo Err_Doc___Find_Click
73
74      Dim DocName As String
75
76      DocName = "Document # Find"
77      DoCmd OpenReport DocName, A_PREVIEW
78
```

```
Exit_Doc___Find_Click:
    Exit Sub

Err_Doc___Find_Click:
    MsgBox Error$
    Resume Exit_Doc___Find_Click

End Sub

Sub Names_Click ()
On Error GoTo Err_Names_Click

    Dim DocName As String

    DocName = "Names"
    DoCmd OpenReport DocName, A_PREVIEW

Exit_Names_Click:
    Exit Sub

Err_Names_Click:
    MsgBox Error$
    Resume Exit_Names_Click

End Sub

Sub Seach_From_Click ()
On Error GoTo Err_Seach_From_Click

    Dim DocName As String

    DocName = "Search From"
    DoCmd OpenReport DocName, A_PREVIEW

Exit_Seach_From_Click:
    Exit Sub

Err_Seach_From_Click:
    MsgBox Error$
    Resume Exit_Seach_From_Click

End Sub

Sub Search_CCs_Click ()
On Error GoTo Err_Search_CCs_Click

    Dim DocName As String

    DocName = "Search CCs"
    DoCmd OpenReport DocName, A_PREVIEW

```

```
181  Exit_Tag_1_Click:
182      Exit Sub
183
184  Err_Tag_1_Click:
185      MsgBox Error$
186      Resume Exit_Tag_1_Click
187
188  End Sub
189
190  Sub Tag_10_Click ()
191  On Error GoTo Err_Tag_10_Click
192
193      Dim DocName As String
194
195      DocName = "List10"
196      DoCmd OpenReport DocName, A_PREVIEW
197
198  Exit_Tag_10_Click:
199      Exit Sub
200
201  Err_Tag_10_Click:
202      MsgBox Error$
203      Resume Exit_Tag_10_Click
204
205  End Sub
206
207  Sub Tag_11_Click ()
208  On Error GoTo Err_Tag_11_Click
209
210      Dim DocName As String
211
212      DocName = "List11"
213      DoCmd OpenReport DocName, A_PREVIEW
214
215  Exit_Tag_11_Click:
216      Exit Sub
217
218  Err_Tag_11_Click:
219      MsgBox Error$
220      Resume Exit_Tag_11_Click
221
222  End Sub
223
224  Sub Tag_12_Click ()
225  On Error GoTo Err_Tag_12_Click
226
227      Dim DocName As String
228
229      DocName = "List12"
230      DoCmd OpenReport DocName, A_PREVIEW
231
```

```
232  Exit_Tag_12_Click:
233      Exit Sub
234
235  Err_Tag_12_Click:
236      MsgBox Error$
237      Resume Exit_Tag_12_Click
238
239  End Sub
240
241  Sub Tag_13_Click ()
242  On Error GoTo Err_Tag_13_Click
243
244      Dim DocName As String
245
246      DocName = "List13"
247      DoCmd OpenReport DocName, A_PREVIEW
248
249  Exit_Tag_13_Click:
250      Exit Sub
251
252  Err_Tag_13_Click:
253      MsgBox Error$
254      Resume Exit_Tag_13_Click
255
256  End Sub
257
258  Sub Tag_14_Click ()
259  On Error GoTo Err_Tag_14_Click
260
261      Dim DocName As String
262
263      DocName = "List14"
264      DoCmd OpenReport DocName, A_PREVIEW
265
266  Exit_Tag_14_Click:
267      Exit Sub
268
269  Err_Tag_14_Click:
270      MsgBox Error$
271      Resume Exit_Tag_14_Click
272
273  End Sub
274
275  Sub Tag_15_Click ()
276  On Error GoTo Err_Tag_15_Click
277
278      Dim DocName As String
279
280      DocName = "List15"
281      DoCmd OpenReport DocName, A_PREVIEW
282
```

```
283  Exit_Tag_15_Click:
284      Exit Sub
285
286  Err_Tag_15_Click:
287      MsgBox Error$
288      Resume Exit_Tag_15_Click
289
290  End Sub
291
292  Sub Tag_16_Click ()
293  On Error GoTo Err_Tag_16_Click
294
295      Dim DocName As String
296
297      DocName = "List16"
298      DoCmd OpenReport DocName, A_PREVIEW
299
300  Exit_Tag_16_Click:
301      Exit Sub
302
303  Err_Tag_16_Click:
304      MsgBox Error$
305      Resume Exit_Tag_16_Click
306
307  End Sub
308
309  Sub Tag_17_Click ()
310  On Error GoTo Err_Tag_17_Click
311
312      Dim DocName As String
313
314      DocName = "List17"
315      DoCmd OpenReport DocName, A_PREVIEW
316
317  Exit_Tag_17_Click:
318      Exit Sub
319
320  Err_Tag_17_Click:
321      MsgBox Error$
322      Resume Exit_Tag_17_Click
323
324  End Sub
325
326  Sub Tag_18_Click ()
327  On Error GoTo Err_Tag_18_Click
328
329      Dim DocName As String
330
331      DocName = "List18"
332      DoCmd OpenReport DocName, A_PREVIEW
333
```

```
334  Exit_Tag_18_Click:
335      Exit Sub
336
337  Err_Tag_18_Click:
338      MsgBox Error$
339      Resume Exit_Tag_18_Click
340
341  End Sub
342
343  Sub Tag_19_Click ()
344  On Error GoTo Err_Tag_19_Click
345
346      Dim DocName As String
347
348      DocName = "List19"
349      DoCmd OpenReport DocName, A_PREVIEW
350
351  Exit_Tag_19_Click:
352      Exit Sub
353
354  Err_Tag_19_Click:
355      MsgBox Error$
356      Resume Exit_Tag_19_Click
357
358  End Sub
359
360  Sub Tag_2_Click ()
361  On Error GoTo Err_Tag_2_Click
362
363      Dim DocName As String
364
365      DocName = "List02"
366      DoCmd OpenReport DocName, A_PREVIEW
367
368  Exit_Tag_2_Click:
369      Exit Sub
370
371  Err_Tag_2_Click:
372      MsgBox Error$
373      Resume Exit_Tag_2_Click
374
375  End Sub
376
377  Sub Tag_20_Click ()
378  On Error GoTo Err_Tag_20_Click
379
380      Dim DocName As String
381
382      DocName = "List20"
383      DoCmd OpenReport DocName, A_PREVIEW
384
```

```
385  Exit_Tag_20_Click:
386      Exit Sub
387
388  Err_Tag_20_Click:
389      MsgBox Error$
390      Resume Exit_Tag_20_Click
391
392  End Sub
393
394  Sub Tag_21_Click ()
395  On Error GoTo Err_Tag_21_Click
396
397      Dim DocName As String
398
399      DocName = "List21"
400      DoCmd OpenReport DocName, A_PREVIEW
401
402  Exit_Tag_21_Click:
403      Exit Sub
404
405  Err_Tag_21_Click:
406      MsgBox Error$
407      Resume Exit_Tag_21_Click
408
409  End Sub
410
411  Sub Tag_22_Click ()
412  On Error GoTo Err_Tag_22_Click
413
414      Dim DocName As String
415
416      DocName = "List22"
417      DoCmd OpenReport DocName, A_PREVIEW
418
419  Exit_Tag_22_Click:
420      Exit Sub
421
422  Err_Tag_22_Click:
423      MsgBox Error$
424      Resume Exit_Tag_22_Click
425
426  End Sub
427
428  Sub Tag_23_Click ()
429  On Error GoTo Err_Tag_23_Click
430
431      Dim DocName As String
432
433      DocName = "List23"
434      DoCmd OpenReport DocName, A_PREVIEW
435
```

```
436  Exit_Tag_23_Click:
437      Exit Sub
438
439  Err_Tag_23_Click:
440      MsgBox Error$
441      Resume Exit_Tag_23_Click
442
443  End Sub
444
445  Sub Tag_24_Click ()
446  On Error GoTo Err_Tag_24_Click
447
448      Dim DocName As String
449
450      DocName = "List24"
451      DoCmd OpenReport DocName, A_PREVIEW
452
453  Exit_Tag_24_Click:
454      Exit Sub
455
456  Err_Tag_24_Click:
457      MsgBox Error$
458      Resume Exit_Tag_24_Click
459
460  End Sub
461
462  Sub Tag_3_Click ()
463  On Error GoTo Err_Tag_3_Click
464
465      Dim DocName As String
466
467      DocName = "List03"
468      DoCmd OpenReport DocName, A_PREVIEW
469
470  Exit_Tag_3_Click:
471      Exit Sub
472
473  Err_Tag_3_Click:
474      MsgBox Error$
475      Resume Exit_Tag_3_Click
476
477  End Sub
478
479  Sub Tag_4_Click ()
480  On Error GoTo Err_Tag_4_Click
481
482      Dim DocName As String
483
484      DocName = "List04"
485      DoCmd OpenReport DocName, A_PREVIEW
486
```

```
487  Exit_Tag_4_Click:
488      Exit Sub
489
490  Err_Tag_4_Click:
491      MsgBox Error$
492      Resume Exit_Tag_4_Click
493
494  End Sub
495
496  Sub Tag_5_Click ()
497  On Error GoTo Err_Tag_5_Click
498
499      Dim DocName As String
500
501      DocName = "List05"
502      DoCmd OpenReport DocName, A_PREVIEW
503
504  Exit_Tag_5_Click:
505      Exit Sub
506
507  Err_Tag_5_Click:
508      MsgBox Error$
509      Resume Exit_Tag_5_Click
510
511  End Sub
512
513  Sub Tag_6_Click ()
514  On Error GoTo Err_Tag_6_Click
515
516      Dim DocName As String
517
518      DocName = "List06"
519      DoCmd OpenReport DocName, A_PREVIEW
520
521  Exit_Tag_6_Click:
522      Exit Sub
523
524  Err_Tag_6_Click:
525      MsgBox Error$
526      Resume Exit_Tag_6_Click
527
528  End Sub
529
530  Sub Tag_7_Click ()
531  On Error GoTo Err_Tag_7_Click
532
533      Dim DocName As String
534
535      DocName = "List07"
536      DoCmd OpenReport DocName, A_PREVIEW
537
```

```
538  Exit_Tag_7_Click:
539      Exit Sub
540
541  Err_Tag_7_Click:
542      MsgBox Error$
543      Resume Exit_Tag_7_Click
544
545  End Sub
546
547  Sub Tag_8_Click ()
548  On Error GoTo Err_Tag_8_Click
549
550      Dim DocName As String
551
552      DocName = "List08"
553      DoCmd OpenReport DocName, A_PREVIEW
554
555  Exit_Tag_8_Click:
556      Exit Sub
557
558  Err_Tag_8_Click:
559      MsgBox Error$
560      Resume Exit_Tag_8_Click
561
562  End Sub
563
564  Sub Tag_9_Click ()
565  On Error GoTo Err_Tag_9_Click
566
567      Dim DocName As String
568
569      DocName = "List09"
570      DoCmd OpenReport DocName, A_PREVIEW
571
572  Exit_Tag_9_Click:
573      Exit Sub
574
575  Err_Tag_9_Click:
576      MsgBox Error$
577      Resume Exit_Tag_9_Click
578
579  End Sub
580
```

CDINFO.DAT

An ASCII file used by Paper Chaser when CDs are used. It is set up in the following manner.
"Yes" implied the case uses CDs. 2 is the Length of the image filename prefix. These are
both on the first line. Each disk is given its own line. The first number is the disk
number, the second is the starting document number and the third is the ending document
number for the CD. The numbers are separated by commas. CDINFO.DAT must reside in the case
directory.

EXAMPLE

YES, 2
1, 00001, 05000
2, 05001, 05799

```
'Option Explicit
'DefInt A-Z

'Allows program to float on top of all programs
'Private Declare Function SetWindowPos Lib "user" (ByVal h%, ByVal hb%,
 ByVal X%, ByVal Y%, ByVal cx%, ByVal cy%, ByVal f%) As Integer
'reads ini files
Private Declare Function GetPrivateProfileString Lib "Kernel" (ByVal lp
ApplicationName As String, lpKeyName As Any, ByVal lpDefault As String,
 ByVal lpReturnedString As String, ByVal nSize As Integer, ByVal lpFile
Name As String) As Integer

Dim ms_FileSavePath$
Dim ms_buttonWithFocus$
Dim ms_DataBaseName$
Const OFN_HIDEREADONLY = &H4&

Private Sub buttonToggle()
   On Error Resume Next
   cmdSave.Enabled = Not cmdSave.Enabled
   cmdScan.Enabled = Not cmdScan.Enabled
   cmdScan90.Enabled = Not cmdScan90.Enabled
   cmdScanSave.Enabled = Not cmdScanSave.Enabled
   cmdSetUp.Enabled = Not cmdSetUp.Enabled
   'reset the focus back to the correct button
   Select Case ms_buttonWithFocus$
      Case "cmdScan"
         cmdScan.SetFocus
      Case "cmdScan90"
         cmdScan90.SetFocus
      Case "cmdScanSave"
         cmdScanSave.SetFocus
   End Select

End Sub

Private Sub chkSaveToDataBase_Click()
   If chkSaveToDataBase.Value = 1 Then
      Call pathInfo 'gets the DB and paths to save images
   Else
      ms_FileSavePath$ = App.Path 'save images to the default path
      ' add '\' to path if needed
       If Right(ms_FileSavePath$, 1) <> "\" Then ms_FileSavePath$ = ms_
FileSavePath$ & "\"
   End If
End Sub

Private Sub cmdSave_Click()
On Error GoTo SaveError
   If wmObject.PageCount = 0 Then Exit Sub
   Dim liPages%
   Call buttonToggle
```

```
   Screen.MousePointer = 11
   Load frmDisplay
   frmDisplay.Show
   If chkAutoDeskew.Value = 1 Then Call DeskewImage
   frmDisplay.Hide
   Unload frmDisplay
   liPages% = wmObject.PageCount
   wmObject.DocumentName = txtPrefix.Text & txtSufix.Text
   '[SaveAsLocal  filename, fpage, cpages, fOverwrite]
   wmObject.SaveAsLocal ms_FileSavePath$ & txtPrefix.Text & txtSufix.Te
xt & ".tif", 1, liPages%, 1
   If chkSaveToDataBase.Value = 1 Then Call dbWrite
   Call numbersUpdate
   ' Close the current document, even if it has been modified but not s
aved.
   wmObject.CloseDoc
   Screen.MousePointer = 0
   Call buttonToggle
Exit Sub
SaveError:
   MsgBox "Error# " & Err.Number & " " & Err.Description
   Resume Next
End Sub


Private Sub cmdScan_Click()
   ms_buttonWithFocus$ = "cmdScan"
   Call buttonToggle
   Dim liPages%
   liPages% = wmObject.PageCount
   liPages% = liPages% + 1
   '[Scan fpage, incrPage, maxPages, flags]
   wmObject.Scan liPages%, 1, -1, 0
   txtBatesENum.Text = Format(Int(txtBatesBNum.Text) + wmObject.PageCou
nt - 1, "0000000")
   Call buttonToggle
End Sub

Private Sub cmdScan90_Click()
   ms_buttonWithFocus$ = "cmdScan90"
   Call buttonToggle
      Dim liPages%
   liPages% = wmObject.PageCount
   liPages% = liPages% + 1
   '[Scan fpage, incrPage, maxPages, flags]
   wmObject.Scan liPages%, 1, -1, 1
   txtBatesENum.Text = Format(Int(txtBatesBNum.Text) + wmObject.PageCou
nt - 1, "0000000")
   Call buttonToggle
End Sub
```

```
Private Sub cmdScanSave_Click()
   ms_buttonWithFocus$ = "cmdScanSave"
   Call buttonToggle
   Screen.MousePointer = 11
   '[Scan fpage, incrPage, maxPages, flags]
   wmObject.Scan 1, 1, 1, 0
   txtBatesENum.Text = txtBatesBNum.Text
   '[SaveAsLocal  filename, fpage, cpages, fOverwrite]
   If chkAutoDeskew.Value = 1 Then Call DeskewImage
   wmObject.DocumentName = txtPrefix.Text & txtSufix.Text
   wmObject.SaveAsLocal ms_FileSavePath$ & txtPrefix.Text & txtSufix.Te
xt & ".tif", 1, 1, 1
   If chkSaveToDataBase.Value = 1 Then Call dbWrite
   Call numbersUpdate
   ' Close the current document, even if it has been modified but not s
aved.
   wmObject.CloseDoc
   Screen.MousePointer = 0
   Call buttonToggle
Exit Sub
SaveError:
   MsgBox "Error# " & Err.Number & " " & Err.Description
   Resume Next
End Sub

Private Sub cmdSetup_Click()
   ' Display the Scanner Setting dialog box
   wmObject.ScanSetup
   Call settingsGet
End Sub

Private Sub dbWrite()
On Error GoTo DBError
'   Open "c:\TESTFILE.TXT" For Append As 1   ' Open file for output.
'   Print #1, txtPrefix.Text & txtSufix.Text & ", "; txtPrefix.Text & t
xtSufix.Text & ".tif, "; txtBatesPre.Text & txtBatesBNum.Text & ", "; t
xtBatesPre.Text & txtBatesENum.Text ' Write data to file.
'   Close #1
   '************************************************
   ''check the DOC NUMBER from Document Info
   ''to see if it has already been used
   Set gdb = OpenDatabase(ms_DataBaseName$)
   strSQL = "SELECT * FROM [Document Info]"
   Set mdsChaser = gdb.OpenRecordset(strSQL, dbOpenDynaset)
   mdsChaser.FindFirst "[Doc Number] = " & "'" & txtSufix.Text & "'"
   If mdsChaser.NoMatch Then
      mdsChaser.AddNew 'write data to database
   Else
      mdsChaser.Edit 'overwrite existing data
   End If
   mdsChaser![Doc Number] = txtSufix.Text
   mdsChaser![Entry Info] = txtPrefix.Text & txtSufix.Text & ".tif"
```

```
   mdsChaser![Beg Bates #] = txtBatesPre.Text & txtBatesBNum.Text
   mdsChaser![End Bates #] = txtBatesPre.Text & txtBatesENum.Text
   mdsChaser.Update
   mdsChaser.Close
   Exit Sub
DBError:
   MsgBox "Error# " & Err.Number & " " & Err.Description
   chkSaveToDataBase.Value = 0
   ms_FileSavePath$ = App.Path
   Exit Sub
End Sub

Private Sub DeskewImage()
On Error Resume Next
If chkAutoDeskew.Value = 0 Then Exit Sub
Dim docPages%
Dim curPage%
   docPages% = wmObject.PageCount
   For curPage% = 1 To docPages%
      frmDisplay.lblPages.Caption = "Deskewing page " & curPage% & " of
 " & docPages% & "."
      frmDisplay.pnlPages.FloodPercent = curPage% / docPages% * 100
      wmObject.PageDeskew curPage%
      DoEvents
   Next curPage%
   frmDisplay.lblPages.Caption = "Saving TIFF file."
End Sub

Private Sub Form_Load()
   'Dim li_OnTop%
   'li_OnTop% = SetWindowPos(Me.hWnd, -1, 0, 0, 0, 0, 1)
   ' Create a Watermark Professional object
   Me.Left = 0
   Me.Top = 0
   Set wmObject = CreateObject("Watermark.Automation")
   wmObject.ShowWindow 2    ' Show Watermark Window
   ms_FileSavePath$ = App.Path
   ' add '\' to path if needed
    If Right(ms_FileSavePath$, 1) <> "\" Then ms_FileSavePath$ = ms_Fil
eSavePath$ & "\"
   Call settingsGet
End Sub

Private Sub Form_Unload(Cancel As Integer)
   ' Exit Watermark, even if the current document has been modified but
 not saved.
   ' Setting wmObject to Nothing causes Visual Basic to unload Watermar
k from memory
   wmObject.Exit
   Set wmObject = Nothing
   Set mdsChaser = Nothing
End Sub
```

```
Private Sub settingsGet()
   Dim sFileName$
   Dim sSection$
   Dim sKeyName$
   Dim sDefault$
   Dim sReturn$
   Dim iReturnSize%
   Dim iReturn%
   sSection$ = "FUJIGINE"
   sKeyName$ = "PageSize"
   sDefault$ = ""
   sReturn$ = Space$(25)
   iReturnSize% = 25
   sFileName$ = "C:\windows\wmpro.ini"
   iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$,
 sDefault$, sReturn$, iReturnSize%, ByVal sFileName$)
   Me.Caption = "Luke's WaterMark Scan Utility       " & sReturn$
End Sub



Public Sub pathInfo()
   On Error GoTo PathError
    Dim mdsPathInfo As Recordset
    Dim strSQL$
    'choose a database with common dialog
    CommonDialog1.CancelError = True
    CommonDialog1.DialogTitle = "Select Database"
    CommonDialog1.Flags = OFN_HIDEREADONLY
    CommonDialog1.Filter = "Database files | *.mdb"
    CommonDialog1.FilterIndex = 1
    CommonDialog1.Action = 1
    ms_DataBaseName$ = CommonDialog1.filename
    'check the database for the path of the images files
    Set gdb = OpenDatabase(ms_DataBaseName$)
    strSQL = "SELECT * FROM Preferences"
    Set mdsPathInfo = gdb.OpenRecordset(strSQL, dbOpenDynaset)
    'set the file save path to it
    ms_FileSavePath$ = mdsPathInfo!ImagePath
    ' add '\' to path if needed
    If Right(ms_FileSavePath$, 1) <> "\" Then ms_FileSavePath$ = ms_Fil
eSavePath$ & "\"
    mdsPathInfo.Close
    Set mdsPathInfo = Nothing
    Exit Sub
PathError:
   If Err = 32755 Then ' cancel button was pressed
      chkSaveToDataBase.Value = 0
      ms_FileSavePath$ = App.Path
      Exit Sub
```

```
   ElseIf Err = 3078 Then
      chkSaveToDataBase.Value = 0
      MsgBox "The database choosen is either not a PaperChaser database
, or it is corrupted"
      ms_FileSavePath$ = App.Path
      Exit Sub
   Else '
      MsgBox "Error# " & Err.Number & " " & Err.Description
      chkSaveToDataBase.Value = 0
      ms_FileSavePath$ = App.Path
      Exit Sub
   End If
End Sub

Public Sub numbersUpdate()
   txtSufix.Text = Format(Int(txtSufix.Text) + 1, "00000")
   txtBatesBNum.Text = Format(Int(txtBatesENum.Text) + 1, "0000000")
   txtBatesENum.Text = Format(Int(txtBatesENum.Text) + 1, "0000000")
End Sub
```

```
frmDisplay - 1

Option Explicit

Private Sub Form_Load()
   Me.Top = (Screen.Height - Me.Height) / 2
   Me.Left = (Screen.Width - Me.Width) / 2
End Sub
```

```
frmWMScan - 1

'Option Explicit
'DefInt A-Z

'Allows program to float on top of all programs
'Private Declare Function SetWindowPos Lib "user" (ByVal h%, ByVal hb%, ByVal X%, ByVal Y%, ByVa
l cx%, ByVal cy%, ByVal f%) As Integer
'reads ini files
Private Declare Function GetPrivateProfileString Lib "Kernel" (ByVal lpApplicationName As String
, lpKeyName As Any, ByVal lpDefault As String, ByVal lpReturnedString As String, ByVal nSize As
Integer, ByVal lpFileName As String) As Integer

Dim ms_FileSavePath$
Dim ms_buttonWithFocus$
Dim ms_DataBaseName$
Const OFN_HIDEREADONLY = &H4&

Private Sub buttonToggle()
   On Error Resume Next
   cmdSave.Enabled = Not cmdSave.Enabled
   cmdScan.Enabled = Not cmdScan.Enabled
   cmdScan90.Enabled = Not cmdScan90.Enabled
   cmdScanSave.Enabled = Not cmdScanSave.Enabled
   cmdSetUp.Enabled = Not cmdSetUp.Enabled
   'reset the focus back to the correct button
   Select Case ms_buttonWithFocus$
      Case "cmdScan"
         cmdScan.SetFocus
      Case "cmdScan90"
         cmdScan90.SetFocus
      Case "cmdScanSave"
         cmdScanSave.SetFocus
   End Select

End Sub

Private Sub chkSaveToDataBase_Click()
   If chkSaveToDataBase.Value = 1 Then
      Call pathInfo 'gets the DB and paths to save images
   Else
      ms_FileSavePath$ = App.Path 'save images to the default path
      ' add '\' to path if needed
       If Right(ms_FileSavePath$, 1) <> "\" Then ms_FileSavePath$ = ms_FileSavePath$ & "\"
   End If
End Sub

Private Sub cmdSave_Click()
On Error GoTo SaveError
   If wmObject.PageCount = 0 Then Exit Sub
   Dim liPages%
   Call buttonToggle
   Screen.MousePointer = 11
   Load frmDisplay
   frmDisplay.Show
   If chkAutoDeskew.Value = 1 Then Call DeskewImage
   frmDisplay.Hide
   Unload frmDisplay
   liPages% = wmObject.PageCount
   wmObject.DocumentName = txtPrefix.Text & txtSufix.Text
   '[SaveAsLocal  filename, fpage, cpages, fOverwrite]
   wmObject.SaveAsLocal ms_FileSavePath$ & txtPrefix.Text & txtSufix.Text & ".tif", 1, liPages%,
1
   If chkSaveToDataBase.Value = 1 Then Call dbWrite
   Call numbersUpdate
   ' Close the current document, even if it has been modified but not saved.
   wmObject.CloseDoc
   Screen.MousePointer = 0
   Call buttonToggle
Exit Sub
SaveError:
```

```
   MsgBox "Error# " & Err.Number & " " & Err.Description
   Resume Next
End Sub


Private Sub cmdScan_Click()
   ms_buttonWithFocus$ = "cmdScan"
   Call buttonToggle
   Dim liPages%
   liPages% = wmObject.PageCount
   liPages% = liPages% + 1
   '[Scan fpage, incrPage, maxPages, flags]
   wmObject.Scan liPages%, 1, -1, 0
   'txtBatesENum.Text = Format(Int(txtBatesBNum.Text) + wmObject.PageCount - 1, "0000000")
   Call buttonToggle
End Sub

Private Sub cmdScan90_Click()
   ms_buttonWithFocus$ = "cmdScan90"
   Call buttonToggle
      Dim liPages%
   liPages% = wmObject.PageCount
   liPages% = liPages% + 1
   '[Scan fpage, incrPage, maxPages, flags]
   wmObject.Scan liPages%, 1, -1, 1
   'txtBatesENum.Text = Format(Int(txtBatesBNum.Text) + wmObject.PageCount - 1, "0000000")
   Call buttonToggle
End Sub

Private Sub cmdScanSave_Click()
   ms_buttonWithFocus$ = "cmdScanSave"
   Call buttonToggle
   Screen.MousePointer = 11
   '[Scan fpage, incrPage, maxPages, flags]
   wmObject.Scan 1, 1, 1, 0
   'txtBatesENum.Text = txtBatesBNum.Text
   '[SaveAsLocal  filename, fpage, cpages, fOverwrite]
   If chkAutoDeskew.Value = 1 Then Call DeskewImage
   wmObject.DocumentName = txtPrefix.Text & txtSufix.Text
   wmObject.SaveAsLocal ms_FileSavePath$ & txtPrefix.Text & txtSufix.Text & ".tif", 1, 1, 1
   If chkSaveToDataBase.Value = 1 Then Call dbWrite
   Call numbersUpdate
   ' Close the current document, even if it has been modified but not saved.
   wmObject.CloseDoc
   Screen.MousePointer = 0
   Call buttonToggle
Exit Sub
SaveError:
   MsgBox "Error# " & Err.Number & " " & Err.Description
   Resume Next
End Sub

Private Sub cmdSetup_Click()
   ' Display the Scanner Setting dialog box
   wmObject.ScanSetup
   Call settingsGet
End Sub

Private Sub dbWrite()
On Error GoTo DBError
'   Open "c:\TESTFILE.TXT" For Append As 1   ' Open file for output.
'   Print #1, txtPrefix.Text & txtSufix.Text & ", "; txtPrefix.Text & txtSufix.Text & ".tif, ";
txtBatesPre.Text & txtBatesBNum.Text & ", "; txtBatesPre.Text & txtBatesENum.Text ' Write data t
o file.
'   Close 01
   '*************************************************
   ''check the DOC NUMBER from Document Info
   ''to see if it has already been used
```

frmWMScan - 3

```
    Set gdb = OpenDatabase(ms_DataBaseName$)
    strSQL = "SELECT * FROM [Document Info]"
    Set mdsChaser = gdb.OpenRecordset(strSQL, dbOpenDynaset)
    mdsChaser.FindFirst "[Doc Number] = " & "'" & txtSufix.Text & "'"
    If mdsChaser.NoMatch Then
       mdsChaser.AddNew 'write data to database
    Else
       mdsChaser.Edit 'overwrite existing data
    End If
    mdsChaser![Doc Number] = txtSufix.Text
    mdsChaser![Entry Info] = txtPrefix.Text & txtSufix.Text & ".tif"
'    mdsChaser![Beg Bates 0] = txtBatesPre.Text & txtBatesBNum.Text
'    mdsChaser![End Bates 0] = txtBatesPre.Text & txtBatesENum.Text
    mdsChaser![Description] = txtDescription.Text
    mdsChaser.Update
    mdsChaser.Close
    Exit Sub
DBError:
    MsgBox "Error# " & Err.Number & " " & Err.Description
    chkSaveToDataBase.Value = 0
    ms_FileSavePath$ = App.Path
    Exit Sub
End Sub

Private Sub DeskewImage()
On Error Resume Next
If chkAutoDeskew.Value = 0 Then Exit Sub
Dim docPages%
Dim curPage%
    docPages% = wmObject.PageCount
    For curPage% = 1 To docPages%
       frmDisplay.lblPages.Caption = "Deskewing page " & curPage% & " of " & docPages% & "."
       frmDisplay.pnlPages.FloodPercent = curPage% / docPages% * 100
       wmObject.PageDeskew curPage%
       DoEvents
    Next curPage%
    frmDisplay.lblPages.Caption = "Saving TIFF file."
End Sub

Private Sub Form_Load()
    'Dim li_OnTop%
    'li_OnTop% = SetWindowPos(Me.hWnd, -1, 0, 0, 0, 0, 1)
    ' Create a Watermark Professional object
    Me.Left = 0
    Me.Top = 0
    Set wmObject = CreateObject("Watermark.Automation")
    wmObject.ShowWindow 2    ' Show Watermark Window
    ms_FileSavePath$ = App.Path
    ' add '\' to path if needed
     If Right(ms_FileSavePath$, 1) <> "\" Then ms_FileSavePath$ = ms_FileSavePath$ & "\"
    Call settingsGet
End Sub

Private Sub Form_Unload(Cancel As Integer)
    ' Exit Watermark, even if the current document has been modified but not saved.
    ' Setting wmObject to Nothing causes Visual Basic to unload Watermark from memory
    wmObject.Exit
    Set wmObject = Nothing
    Set mdsChaser = Nothing
End Sub

Private Sub settingsGet()
    Dim sFileName$
    Dim sSection$
    Dim sKeyName$
    Dim sDefault$
    Dim sReturn$
    Dim iReturnSize%
    Dim iReturn%
```

```
   sSection$ = "FUJIGINE"
   sKeyName$ = "PageSize"
   sDefault$ = ""
   sReturn$ = Space$(25)
   iReturnSize% = 25
   sFileName$ = "C:\windows\wmpro.ini"
   iReturn% = GetPrivateProfileString(ByVal sSection$, ByVal sKeyName$, sDefault$, sReturn$, iRe
turnSize%, ByVal sFileName$)
   Me.Caption = "Luke's WaterMark Scan Utility       " & sReturn$
End Sub



Public Sub pathInfo()
   On Error GoTo PathError
    Dim mdsPathInfo As Recordset
    Dim strSQL$
    'choose a database with common dialog
    CommonDialog1.CancelError = True
    CommonDialog1.DialogTitle = "Select Database"
    CommonDialog1.Flags = OFN_HIDEREADONLY
    CommonDialog1.Filter = "Database files | *.mdb"
    CommonDialog1.FilterIndex = 1
    CommonDialog1.Action = 1
    ms_DataBaseName$ = CommonDialog1.filename
    'check the database for the path of the images files
    Set gdb = OpenDatabase(ms_DataBaseName$)
    strSQL = "SELECT * FROM Preferences"
    Set mdsPathInfo = gdb.OpenRecordset(strSQL, dbOpenDynaset)
    'set the file save path to it
    ms_FileSavePath$ = mdsPathInfo!ImagePath
    ' add '\' to path if needed
    If Right(ms_FileSavePath$, 1) <> "\" Then ms_FileSavePath$ = ms_FileSavePath$ & "\"
    mdsPathInfo.Close
    Set mdsPathInfo = Nothing
    Exit Sub
PathError:
   If Err = 32755 Then ' cancel button was pressed
      chkSaveToDataBase.Value = 0
      ms_FileSavePath$ = App.Path
      Exit Sub
   ElseIf Err = 3078 Then
      chkSaveToDataBase.Value = 0
      MsgBox "The database choosen is either not a PaperChaser database, or it is corrupted"
      ms_FileSavePath$ = App.Path
      Exit Sub
   Else '
      MsgBox "Error# " & Err.Number & " " & Err.Description
      chkSaveToDataBase.Value = 0
      ms_FileSavePath$ = App.Path
      Exit Sub
   End If
End Sub

Public Sub numbersUpdate()
   txtSufix.Text = Format(Int(txtSufix.Text) + 1, "00000")
   'txtBatesBNum.Text = Format(Int(txtBatesENum.Text) + 1, "0000000")
   'txtBatesENum.Text = Format(Int(txtBatesENum.Text) + 1, "0000000")
End Sub
```

```
WMSCAN1 - 1

Option Explicit
Global wmObject As Object   ' The object handle to Watermark
```

```
frmDisplay - 1

Option Explicit

Private Sub Form_Load()
   Me.Top = (Screen.Height - Me.Height) / 2
   Me.Left = (Screen.Width - Me.Width) / 2
End Sub
```

```
frmOCR - 1

' Copyright (C) 1995,1996  Luke Spence
' Last modified on 05/01/96
Option Explicit
DefInt A-Z
Private Declare Function FindWindow% Lib "user" (ByVal lpClassName As Any, ByVal lpCaption As An
y)
'Declare Sub SetCursorPos Lib "User" (ByVal x As Integer, ByVal y As Integer)
'There's no need to reset mouse position with OmniPage Lite
Dim SetUp
Dim programCaption$
Dim programPath$
Dim FName$
Dim DocFName$
Dim filePath$
Dim fileToGet$
Dim numOfFiles%
Dim x%
Dim numSelected%
Dim percentDone%
Dim procCount%
Dim test%

Private Sub cmdExit_Click()
  End
End Sub

Private Sub cmdProcess_Click()
  Dim iLoop%
  On Error Resume Next
  If File1.filename = "" Then
    MsgBox "Please choose a file.", 0, "No file selected."
    Exit Sub
  End If
  If Right(File1.Path, 1) = "\" Then
    filePath = File1.Path
  Else
    filePath = File1.Path & "\"
  End If
  numOfFiles% = File1.ListCount
  For iLoop% = 0 To numOfFiles% - 1
    If File1.Selected(iLoop%) Then numSelected% = numSelected% + 1
  Next iLoop%
  percentDone% = 100 \ numSelected%
  For iLoop% = 0 To numOfFiles% - 1
    fileToGet$ = File1.List(iLoop%)
    If File1.Selected(iLoop%) Then
      FName = filePath & fileToGet
      DocFName = Left$(fileToGet, Len(fileToGet  - 3))
      procCount% = procCount% + 1
      pnlDisplay.Caption = "Processing " & procCount% & " of " & numSelected%
      Call Process
      DoEvents
      pnlPercentComplete.FloodPercent = pnlPercentComplete.FloodPercent + percentDone
    End If
  Next iLoop%
  pnlDisplay.Caption = "Finished processing at " & Time
  pnlPercentComplete.FloodPercent = 0
  procCount% = 0
  numSelected% = 0
  File1.Refresh
End Sub

Private Sub Dir1_Change()
  File1.Path = Dir1.Path

End Sub

Private Sub Drive1_Change()
  On Error Resume Next
```

```
frmOCR - 2

  Dir1.Path = Drive1.Drive
End Sub

Private Sub Form_Load()
   On Error Resume Next
   Top = 100
   Left = Screen.Width - frmOCR.Width - 100
   Call infoGet
   x% = Shell(programPath$, 4)
   frmOCR.Show
   Dir1.Path = "C:\"
End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, x As Single, y As Single)
  x = 1
  y = 1
End Sub

Private Sub infoGet()
   On Error GoTo iniError
   Open "lukesocr.ini" For Input As #1
   Line Input #1, programCaption$ ' Get complete line.
   Line Input #1, programPath$ ' Get complete line.
   Close #1
   Exit Sub
iniError:
   MsgBox "Error retrieving information from 'lukesocr.ini'"
End Sub

Private Sub mouseHome()
   ' need a routine to move mouse pointer to bottom corner
   ' of the screen so that WordScan title bar doesn't change
   ' if it does VB program can't recognize WordScan program
   'SetCursorPos Screen.Width, Screen.Height
End Sub

Private Sub Process()
   AppActivate programCaption$
   x% = Shell(programPath$ & " " & FName$, 4)
   Call mouseHome
   SendKeys "%F", True  'ALT F
   SendKeys "R", True   'R
   SendKeys "A", True   'A
   test% = 0
   Do While test% = 0
      x% = DoEvents()
      test% = FindWindow(0&, "Save As")
   Loop
   SendKeys DocFName$
   SendKeys "{Enter}"
   test% = 0
   Call mouseHome
   SendKeys "{Enter}" ' to overwrite if file already exists
   Do While test% = 0
      x% = DoEvents()
      test% = FindWindow(0&, programCaption$)
   Loop
   AppActivate "Luke's Automated OCR'ing Utility"
End Sub
```

```
frmDepoView - 1

Option Explicit
DefInt A-Z
'''arrays
   Dim page() As Long
   Dim finds() As Integer
'''long
   Dim position&
'''integers
   Dim pageNumber%
   Dim numOfPages%
   Dim startPosition%
   Dim f%
   Dim foundcount%
   Dim FirstPageEOPMarker%
   Dim iPageNumShown%
'''strings
   Dim pathOfDepos$
   Dim fileSource$
   Dim tmpLocations$
   Dim new_Line$
   Dim endOfPage$
   Dim search$
   Dim temp$
   Dim nameOfDeposed$

Private Sub buttonReset()
   On Error Resume Next
   ' reset page number buttons
   Select Case pageNumber%
   Case 0              ' no pages loaded
      cmdPagePrev.Enabled = False
      cmdPageFirst.Enabled = False
      cmdPageNext.Enabled = False
      cmdPageLast.Enabled = False
   Case 1              ' first page
      cmdPagePrev.Enabled = False
      cmdPageFirst.Enabled = False
      cmdPageNext.Enabled = True
      cmdPageLast.Enabled = True
   Case numOfPages     ' last page
      cmdPagePrev.Enabled = True
      cmdPageFirst.Enabled = True
      cmdPageNext.Enabled = False
      cmdPageLast.Enabled = False
   Case Else           ' all other pages
      cmdPagePrev.Enabled = True
      cmdPageFirst.Enabled = True
      cmdPageNext.Enabled = True
      cmdPageLast.Enabled = True
   End Select
   Select Case numOfPages%    ' if only one page
   Case 1
      cmdPageNext.Enabled = False
      cmdPageLast.Enabled = False
   End Select
   ' reset find word buttons
   Select Case foundcount%
   Case 0              ' no words found
      cmdWordPrev.Enabled = False
      cmdWordFirst.Enabled = False
      cmdWordNext.Enabled = False
      cmdWordLast.Enabled = False
   Case 1              ' only 1 occurance of word
      cmdWordPrev.Enabled = False
      cmdWordFirst.Enabled = False
      cmdWordNext.Enabled = False
      cmdWordLast.Enabled = False
   Case Else           ' more than one occurance
      Select Case finds(f%)
```

frmDepoView - 2

```
      Case finds(1)              ' first occurance of word
         cmdWordPrev.Enabled = False
         cmdWordFirst.Enabled = False
         cmdWordNext.Enabled = True
         cmdWordLast.Enabled = True
      Case finds(foundcount%) ' last occurance of word
         cmdWordPrev.Enabled = True
         cmdWordFirst.Enabled = True
         cmdWordNext.Enabled = False
         cmdWordLast.Enabled = False
      Case Else                  ' all other words
         cmdWordPrev.Enabled = True
         cmdWordFirst.Enabled = True
         cmdWordNext.Enabled = True
         cmdWordLast.Enabled = True
      End Select
   End Select
   ' reset word buttons if no depo is loaded
   Select Case fileSource$
   Case ""
      cmdCopy.Enabled = False
      cmdWordFind.Enabled = False
      cmdWordPrev.Enabled = False
      cmdWordFirst.Enabled = False
      cmdWordNext.Enabled = False
      cmdWordLast.Enabled = False
   Case Else
      cmdCopy.Enabled = True
      cmdWordFind.Enabled = True
   End Select
End Sub

Private Sub cmdCopy_Click()
   Dim cbDepoInfo$
   cbDepoInfo$ = "From the deposition of " & nameOfDeposed$ & ", page # " & iPageNumShown% & new
_Line$
   Clipboard.Clear
   Clipboard.SetText new_Line$ & cbDepoInfo$ & txtCopyPaste.SelText & new_Line$
End Sub

Private Sub cmdPageFirst_Click()
   On Error Resume Next
   pageNumber% = 1
   Call pageDisplay
End Sub

Private Sub cmdPageLast_Click()
   On Error Resume Next
   pageNumber% = numOfPages%
   Call pageDisplay
End Sub

Private Sub cmdPageNext_Click()
   On Error Resume Next
   pageNumber% = pageNumber% + 1
   Call pageDisplay
End Sub

Private Sub cmdPagePrev_Click()
   On Error Resume Next
   pageNumber% = pageNumber% - 1
   Call pageDisplay
End Sub

Private Sub cmdWordFind_Click()
   On Error Resume Next
   Dim pageDisplayed%
   pageDisplayed% = pageNumber%
   pctDisplay.Visible = True
```

```
frmDepoView - 3

   searchS = InputBox$("Please enter the word you wish to search for.", "Word Search", "")
   If searchS = "" Then Exit Sub
   ' reset variables
   f% = 0
   foundcount% = 0
   Call wordFind
   If f% < 1 Then
      pageNumber% = pageDisplayed%
      MsgBox "The word '" & searchS & "' was not found.", 0, "Not Found."
      Exit Sub
   End If
   pageNumber% = finds(f%)
   Call pageDisplay
End Sub

Private Sub cmdWordFirst_Click()
   On Error Resume Next
   f% = 1
   pageNumber% = finds(f%)
   Call pageDisplay
End Sub

Private Sub cmdWordLast_Click()
   On Error Resume Next
   f% = foundcount%
   pageNumber% = finds(f%)
   Call pageDisplay
End Sub

Private Sub cmdWordNext_Click()
   On Error Resume Next
   f% = f% + 1
   pageNumber% = finds(f%)
   Call pageDisplay
End Sub

Private Sub cmdWordPrev_Click()
   On Error Resume Next
   f% = f% - 1
   pageNumber% = finds(f%)
   Call pageDisplay
End Sub

Private Sub deposedName()
   Dim firstPageS
   Dim posMarker%
   Dim posEOL1%
   Dim posEOL2%
   Dim posEOL3%
   tempS = ""
   Open fileSourceS For Binary Access Read As #1
   Seek #1, 1
   firstPageS = InputS(1096, 1)
   Close #1
   posMarker% = InStr(1, firstPageS, "deposition of", 1)
   posEOL1% = InStr(posMarker%, firstPageS, new_LineS)
   posEOL2% = InStr(posEOL1% + 1, firstPageS, new_LineS)
   posEOL3% = InStr(posEOL2% + 1, firstPageS, new_LineS)
   tempS = Mid$(firstPageS, (posMarker% + 13), (posEOL1% - (posMarker% - 13)))
   If Len(Trim(tempS)) > 3 Then
      nameOfDeposedS = fnMakeAlpha(tempS)
   Else
      tempS = Mid$(firstPageS, posEOL1%, (posEOL2% - posEOL1%))
      nameOfDeposedS = fnMakeAlpha(tempS)
      If Trim(nameOfDeposedS) = "" Then
         tempS = Mid$(firstPageS, posEOL2%, (posEOL3% - posEOL2%))
         nameOfDeposedS = fnMakeAlpha(tempS)
      End If
   End If
```

```
frmDepoView - 4

   nameOfDeposed$ = Trim(nameOfDeposed$)
End Sub

Private Sub depoSelect()
   On Error Resume Next
   frmDepoView.Dialogue1.CancelError = False
   frmDepoView.Dialogue1.DialogTitle = "Open Deposition"
   frmDepoView.Dialogue1.Flags = OFN_HIDEREADONLY
   frmDepoView.Dialogue1.InitDir = pathOfDepos$
   frmDepoView.Dialogue1.Filter = "All files (*.*)|*.*"
   frmDepoView.Dialogue1.FilterIndex = 1
   frmDepoView.Dialogue1.Action = 1
   fileSource$ = frmDepoView.Dialogue1.filename
End Sub

Private Sub determineFormat()
   On Error Resume Next
   Dim char$
   Dim posFound%
   Open fileSource$ For Binary Access Read As #1
   char$ = Input$(5120, 1)
   Close #1
   posFound% = InStr(char$, endOfPage$)
   If posFound% = 0 Then
      Call pagePositionsAmicus
   Else
      Call pagePositionsAscii
   End If
End Sub

Private Function fnGetFirstLine(pageOfText As String) As String
   Dim eolPos%
   Dim sFirstLineOfText$
   eolPos% = InStr(pageOfText, new_Line$)
   sFirstLineOfText$ = Mid$(pageOfText, 1, eolPos%)
   fnGetFirstLine = sFirstLineOfText$
End Function

Private Function fnGetLastLine(pageOfText As String) As String
   Dim eolPos%
   'remove last eol marker
   pageOfText$ = Mid$(pageOfText$, 1, (Len(pageOfText$) - 2))
   'remove all other eol markers & whittle down pageOfText
   Do While InStr(pageOfText$, new_Line$)
      eolPos% = InStr(pageOfText, new_Line$)
      pageOfText$ = Mid$(pageOfText, eolPos% + 1)
   Loop
   fnGetLastLine = pageOfText$
End Function

Private Function fnMakeAlpha(firstPageOfDepo As String) As String
   Dim char$
   Dim alpha$
   Dim iloop%
   For iloop% = 1 To Len(firstPageOfDepo$)
      char$ = Mid$(firstPageOfDepo$, iloop%, 1)
      Select Case Asc(char$)
      Case 32, 65 To 90, 97 To 122
         alpha$ = alpha$ & char$
      End Select
   Next iloop%
   fnMakeAlpha = alpha$
End Function

Private Function fnMakeNumeric(alphaNumeric As String) As Integer
   Dim char$
   Dim numeric%
   Dim iloop%
   For iloop% = 1 To Len(alphaNumeric$)
```

```
frmDepoView - 5

      char$ = Mid$(alphaNumeric$, iloop%, 1)
      Select Case Asc(char$)
      Case 48 To 57
         numeric% = numeric% & CInt(char$)
      End Select
   Next iloop%
   fnMakeNumeric = numeric%
End Function

Private Sub Form_Load()
   On Error Resume Next
   frmDepoView.Left = (Screen.Width - frmDepoView.Width) / 2
   frmDepoView.Top = (Screen.Height - frmDepoView.Height) / 2
   Call buttonReset
   Call screenSize
   pathOfDepos$ = Command$
   If pathOfDepos$ = "" Then
      pathOfDepos$ = App.Path
   Else
      ' here we want code to replace images/ with depos/
      pathOfDepos$ = Left$(pathOfDepos$, (InStr(1, pathOfDepos$, "Images", 1) - 1)) & "Depos\"
   End If
   endOfPage$ = Chr$(12)
   new_Line$ = Chr$(13)
End Sub

Private Sub Form_Resize()
   Call screenSize
End Sub

Private Sub getPageNumber()
   On Error Resume Next
   Dim iValidSearch%
   Dim iFirstLine%
   Dim iLastLine%
   Dim errCode%
   Dim sFirstLine$
   Dim sLastLine$
   Dim pageFromDepo$
   pageFromDepo$ = txtCopyPaste.Text
   errCode% = 0
   sFirstLine$ = fnGetFirstLine(pageFromDepo$)
   sLastLine$ = fnGetLastLine(pageFromDepo$)
   iFirstLine% = fnMakeNumeric(sFirstLine$)
   iLastLine% = fnMakeNumeric(sLastLine$)
   ' check for page numbers with word 'page' eg "Page 148"
   iValidSearch% = InStr(1, sFirstLine$, "page", 1)
   If iValidSearch% > 0 Then
      If iFirstLine% > 0 Then
         iPageNumShown% = iFirstLine%
         Exit Sub
      End If
   End If
   iValidSearch% = InStr(1, sLastLine$, "page", 1)
   If iValidSearch% > 0 Then
      If iLastLine% > 0 Then
         iPageNumShown% = iLastLine%
         Exit Sub
      End If
   End If
   ' check for amicus style number eg "00148"
   On Error GoTo amicusLastLine
   Select Case Len(Trim(sFirstLine$))
   Case 4, 5
      iPageNumShown% = CInt(Trim(sFirstLine$))
      Exit Sub
   End Select
amicusLastLine:
   On Error GoTo genericNumber
```

frmDepoView - 6

```
   Select Case Len(Trim(sLastLine$))
   Case 4, 5
      iPageNumShown% = CInt(Trim(sLastLine$))
      Exit Sub
   End Select
genericNumber: ' check for generic number eg "148"
   On Error Resume Next
   Select Case Len(Trim(sFirstLine$))
   Case 1, 2, 3
      If iFirstLine% > 0 Then
         iPageNumShown% = iFirstLine%
         Exit Sub
      End If
   End Select
   Select Case Len(Trim(sLastLine$))
   Case 1, 2, 3
      If iLastLine% > 0 Then
         iPageNumShown% = iLastLine%
         Exit Sub
      End If
   End Select
   iPageNumShown% = 0
End Sub

Private Sub mnuEditCopy_Click()
   cmdCopy_Click
End Sub

Private Sub mnuEditFind_Click()
   cmdWordFind_Click
End Sub

Private Sub mnuFileExit_Click()
   End
End Sub

Private Sub mnuFileOpen_Click()
   On Error Resume Next
   'reset a few variables
   pageNumber% = 1
   search$ = ""
   pctDisplay.Visible = True
   Call depoSelect
   If frmDepoView.Dialogue1.filename = "" Then Exit Sub
   'Call determineFormat
   Call pagePositionsAscii 'or pagePositionsAmicus
   Call pageLocations
   Call deposedName
   Call pageDisplay
End Sub

Private Sub mnuFilePrint_Click()
   On Error Resume Next
   Dim x%
   Dim pos&
   Dim EOPMarker&
   Dim pageList1$
   Dim found%
   Dim currentLine$
   Dim pageList1Trim$
   Printer.FontBold = True
   Printer.FontSize = 12
   Printer.FontName = "Courier"
   Printer.FontSize = 12
   For x% = 1 To numOfPages%
      Printer.Print " ": Printer.Print " ":  Printer.Print " ": Printer.Print " "
      Printer.Print " ": Printer.Print " ":  Printer.Print " ": Printer.Print " "
      pageNumber% = x%
      pos& = 1
```

```
frmDepoView - 7

      If pageNumber% = 1 Then
         If FirstPageEOPMarker% = 1 Then pos& = 2
      Else
         pos& = page(pageNumber - 1) + 1
      End If
      EOPMarker& = page(pageNumber%)
      Open fileSource$ For Binary Access Read As #1
      Seek #1, pos&
      pageList1$ = Input$((EOPMarker& - pos&), 1)
      Close #1
      Do While Len(pageList1$) <> 0
         found% = InStr(pageList1$, new_Line$)
         currentLine$ = Mid$(pageList1$, 1, (found% - 1))
         If Trim(currentLine$) <> "" Then
            pageList1Trim$ = pageList1Trim$ & currentLine$ & Chr$(13) & Chr$(10)
         End If
         pageList1$ = Mid$(pageList1$, (found% + 2))
      Loop
      Printer.Print pageList1Trim$
      Printer.NewPage
      pageList1Trim$ = ""
   Next x%
   Printer.EndDoc
End Sub

Private Sub mnuFilePrinterCondensed_Click()
   On Error Resume Next
   Dim x%
   Dim pos&
   Dim EOPMarker&
   Dim pageList1$
   Dim found%
   Dim currentLine$
   Dim pageList1Trim$
   Dim liPrintPosX%
   Dim liPrintPosY%
   Dim liQuadrant%
   liQuadrant% = 1
   Printer.FontBold = True
   Printer.FontSize = 7
   Printer.FontName = "Courier"
   Printer.FontSize = 7
   For x% = 1 To numOfPages%
      pageNumber% = x%
      pos& = 1
      If pageNumber% = 1 Then
         If FirstPageEOPMarker% = 1 Then pos& = 2
      Else
         pos& = page(pageNumber - 1) + 1
      End If
      EOPMarker& = page(pageNumber%)
      Open fileSource$ For Binary Access Read As #1
      Seek #1, pos&
      pageList1$ = Input$((EOPMarker& - pos&), 1)
      Close #1
      If liQuadrant% = 1 Then
         liPrintPosX% = 0
         liPrintPosY% = 0
         liQuadrant% = 2
      ElseIf liQuadrant% = 2 Then
         liPrintPosX% = 0
         liPrintPosY% = Printer.ScaleHeight / 2
         liQuadrant% = 3
      ElseIf liQuadrant% = 3 Then
         liPrintPosX% = Printer.ScaleWidth / 2
         liPrintPosY% = 0
         liQuadrant% = 4
      ElseIf liQuadrant% = 4 Then
         liPrintPosX% = Printer.ScaleWidth / 2
```

```
frmDepoView - 8

        liPrintPosY% = Printer.ScaleHeight / 2
        liQuadrant% = 1
      End If

      Printer.CurrentY = liPrintPosY%
      Printer.Print " ": Printer.Print " "
      Do While Len(pageList1$) <> 0
         found% = InStr(pageList1$, new_Line$)
         currentLine$ = Mid$(pageList1$, 1, (found% - 1))
         If Trim(currentLine$) <> "" Then
            'pageList1Trim$ = pageList1Trim$ & currentLine$ & Chr$(13) & Chr$(10)
            Printer.CurrentX = liPrintPosX%
            Printer.Print currentLine$ & new_Line$;
         End If
         pageList1$ = Mid$(pageList1$, (found% + 2))
      Loop
      pageList1Trim$ = ""
      If x% Mod 4 = 0 Then
'        Printer.Line ((Printer.ScaleWidth / 2), 0)-((Printer.ScaleWidth / 2), Printer.ScaleHei
ght)
'        Printer.Line (0, (Printer.ScaleHeight / 2))-(Printer.ScaleWidth, (Printer.ScaleHeight
/ 2))
        Printer.NewPage
      End If
   Next x%

   Printer.EndDoc

End Sub

Private Sub mnuFilePrinterSetup_Click()
    Dim CancelFlag As Integer

    CancelFlag = True
    On Error Resume Next
    Dialogue1.CancelError = True
    Dialogue1.Flags = PD_PRINTSETUP
    Dialogue1.Action = 5
    If (Err = 0) Then
        CancelFlag = False
    End If
    If (CancelFlag = True) Then Exit Sub

End Sub

Private Sub mnuHelpAbout_Click()
   Load frmAbout
  frmAbout.Show
End Sub

Private Sub pageDisplay()
   On Error Resume Next
   Dim found%
   Dim pos&
   Dim currentLine$
   Dim EOPMarker&
   Dim pageList1$
   Dim pageList1Trim$
   pos& = 1
   If pageNumber% = 1 Then
      If FirstPageEOPMarker% = 1 Then pos& = 2
   Else
      pos& = page(pageNumber - 1) + 1
   End If
   EOPMarker& = page(pageNumber%)
   pctDisplay.Cls
   txtCopyPaste.Clear
   Open fileSource$ For Binary Access Read As #1
   Seek #1, pos&
```

```
frmDepoView - 9

   pageList1$ = Input$((EOPMarker& - pos&), 1)
   Close #1
   Do While Len(pageList1$) <> 0
      found% = InStr(pageList1$, new_Line$)
      currentLine$ = Mid$(pageList1$, 1, (found% - 1))
      If Trim(currentLine$) <> "" Then
         If search$ = "" Then
            pctDisplay.Print currentLine$
            pageList1Trim$ = pageList1Trim$ & currentLine$ & Chr$(13) & Chr$(10)
         Else
            pageList1Trim$ = pageList1Trim$ & currentLine$ & Chr$(13) & Chr$(10)
            If InStr(1, currentLine$, search$, 1) Then
               pctDisplay.Print Mid$(currentLine$, 1, (InStr(1, currentLine$, search$, 1)) - 1);
               pctDisplay.ForeColor = &HFF      ' red
               pctDisplay.Print Mid$(currentLine$, (InStr(1, currentLine$, search$, 1)), Len(sea
rch$));
               pctDisplay.ForeColor = &H0 ' black
               pctDisplay.Print Mid$(currentLine$, ((InStr(1, currentLine$, search$, 1)) + Len(s
earch$)))
            Else
               pctDisplay.Print currentLine$
            End If
         End If
      End If
      pageList1$ = Mid$(pageList1$, (found% + 2))
   Loop
   txtCopyPaste.Text = pageList1Trim$
   If InStr(txtCopyPaste.Text, search$, 1) Then
      pctScroll.Visible = True
      txtCopyPaste.Visible = False
   Else
      pctScroll.Visible = False
      txtCopyPaste.Visible = True
   End If
   If search$ = "" Then
      pctScroll.Visible = False
      txtCopyPaste.Visible = True
   End If
   Call buttonReset
   Call getPageNumber
   Me.Caption = "Deposition of " & nameOfDeposed$ & "        Page # " & iPageNumShown%
End Sub

Private Sub pageLocations()
   Dim x%
   Dim found%
   ReDim page(1 To numOfPages%)
   x% = 1
   Do While Len(eopLocations$) > 0
      found% = InStr(eopLocations$, " ")
      If found% <> 0 Then
         page(x%) = Int(Mid(eopLocations$, 1, found%))
         eopLocations$ = Mid$(eopLocations$, (found% + 1))
         x% = x% + 1
      Else
         Exit Do
      End If
   Loop
End Sub

Private Sub pagePositionsAmicus()
MsgBox "Currently usupported file format.", 64, "Unsupported format. "
Exit Sub
'   On Error Resume Next
'   Dim firstPageNumber%
'   Dim nextPageNumber%
'   Dim amicusPageNumber$
'   Dim trimChar$
'   Dim char$
```

```
frmDepoView - 10

'   Dim zeros$
'   zeros$ = "0000"
'   numOfPages% = 0
'   Open fileSource$ For Input As #1
'      Line Input #1, char$
'      trimChar$ = Trim(char$)
'      firstPageNumber% = CInt(trimChar$)
'   Close #1
'   'loop
'      nextPageNumber% = nextPageNumber% + 1
'      amicusPageNumber$ = Right$((zeros$ & CStr(nextPageNumber%)), 4)
'      'search for amicusPageNumber
'
'   Open fileSource$ For Binary Access Read As #1
'   Do Until EOF(1)           ' find the number of end of page markers
'      char$ = Input$(32768, 1)
'      posFound% = InStr(char$, amicusPageNumber$)
'      If posFound% <> 0 Then
'         numOfPages% = numOfPages% + 1
'         eopLocations$ = eopLocations$ & (offsetPosition& + posFound%) & " "
'      End If
'      offsetPosition& = offsetPosition& + 32768
'   Loop
'   Close #1
'
'
'
'   Seek #1, 1 ' make adjustment if 1st page starts with EOP marker
'   If Input$(1, 1) = endOfPage$ Then
'      numOfPages% = numOfPages% - 1
'      eopLocations$ = Mid$(eopLocations$, 3)
'      FirstPageEOPMarker% = 1' if first page starts with an EOP marker
'   End If
'   Seek #1, LOF(1)  ' make adjustment if last page has no EOF
'   If Input$(1, 1) <> endOfPage$ Then
'      numOfPages% = numOfPages% + 1
'      eopLocations$ = eopLocations$ & FileLen(fileSource$) & " "
'   End If
'
'
'
'   Me.Caption = Me.Caption & " " & numOfPages%
End Sub

Private Sub pagePositionsAscii()
   On Error Resume Next
   Dim char$
   Dim posFound%
   Dim offsetPosition&
   Dim lastCharPosition&
   numOfPages% = 0
   FirstPageEOPMarker% = 0
   lastCharPosition& = FileLen(fileSource$)
   Open fileSource$ For Binary Access Read As #1
   Do Until EOF(1)           ' find the number of end of page markers
      char$ = Input$(128, 1)
      posFound% = InStr(char$, endOfPage$)
      If posFound% <> 0 Then
         numOfPages% = numOfPages% + 1
         eopLocations$ = eopLocations$ & (offsetPosition& + posFound%) & " "
      End If
      offsetPosition& = offsetPosition& + 128
   Loop
   Seek #1, 1 ' make adjustment if 1st page starts with EOP marker
   If Input$(1, 1) = endOfPage$ Then
      numOfPages% = numOfPages% - 1
      eopLocations$ = Mid$(eopLocations$, 3)
      FirstPageEOPMarker% = 1 ' if first page starts with an EOP marker
   End If
```

```
frmDepoView - 11

   Seek #1, LOF(1)   ' make adjustment if last page has no EOF
   If Input$(1, 1) <> endOfPage$ Then
      numOfPages% = numOfPages% + 1
      eopLocations$ = eopLocations$ & FileLen(fileSource$) & " "
   End If
   Close #1
End Sub

Private Sub pctDisplay_Click()
   txtCopyPaste.Visible = True
   pctScroll.Visible = False
   txtCopyPaste.SetFocus
End Sub

Private Sub screenSize()
   On Error Resume Next
   pctScroll.Top = cmdWordPrev.Height + cmdWordPrev.Top + 10
   pctScroll.Left = 10
   pctScroll.Width = frmDepoView.ScaleWidth - 50
   pctScroll.Height = frmDepoView.ScaleHeight - cmdPagePrev.Height - 50
   VScroll1.Top = 0
   VScroll1.Left = pctScroll.Width - VScroll1.Width
   VScroll1.Height = pctScroll.Height
   VScroll1.Max = 100
   VScroll1.LargeChange = 33
   VScroll1.SmallChange = 16
   pctDisplay.Top = 0
   pctDisplay.Left = 0
   pctDisplay.Width = pctScroll.Width - VScroll1.Width - 40
   pctDisplay.Height = Screen.Height * 2
   txtCopyPaste.Top = pctScroll.Top
   txtCopyPaste.Left = pctScroll.Left
   txtCopyPaste.Width = pctScroll.Width
   txtCopyPaste.Height = pctScroll.Height
End Sub

Private Sub VScroll1_Change()
   'pctDisplay.Top = -VScroll1.Value
   pctDisplay.Top = -(VScroll1.Value / 100) * ScaleHeight
End Sub

Private Sub wordFind()
   On Error Resume Next
   Dim x%
   Dim lastPageNumber%
   Dim found%
   Dim lineFromFile$
   Dim tempFinds$
   lastPageNumber% = 0
   temp$ = ""
   Open fileSource$ For Input As #1
   If Input$(1, 1) = endOfPage$ Then
      pageNumber% = 0
   Else
      pageNumber% = 1
   End If
   Close #1
   Open fileSource$ For Input As #1
   Do Until EOF(1)
      Line Input #1, lineFromFile$
      If InStr(lineFromFile$, endOfPage$) Then pageNumber% = pageNumber% + 1
      If InStr(1, lineFromFile$, search$, 1) Then
         If pageNumber% <> lastPageNumber% Then
            lastPageNumber% = pageNumber%
            temp$ = temp$ & pageNumber% & " "
         End If
      End If
   Loop
   Close #1
```

frmDepoView - 12

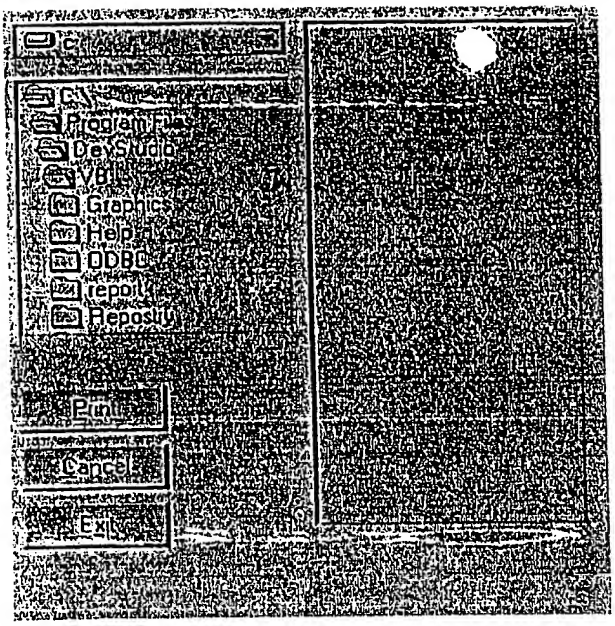
```
   tempFinds$ = temp$
   ' determine the number of numbers in this string then redim an array
   ' to hold each individual number
   If temp$ = "" Then Exit Sub ' no match found
   found% = 0
   Do While Len(temp$) > 0
      found% = InStr(temp$, " ")
      If found% <> 0 Then
         foundcount% = foundcount% + 1
         temp$ = Mid$(temp$, (found% + 1))
      Else
         Exit Do
      End If
   Loop
   ' now extract each page # and put them into into the Finds() array
   ' use Finds() array to bounce around from page to page
   ReDim finds(foundcount% + 1)
   temp$ = tempFinds$
   For x% = 1 To foundcount%
      found% = InStr(temp$, " ")
      finds(x%) = Int(Mid(temp$, 1, (found%)))
      temp$ = Mid$(temp$, (found% + 1))
   Next x%
   f% = 1
End Sub
```

```
frmIntro - 1

Option Explicit
DefInt A-Z

Private Sub displayText()
   FontSize = 42
   frmIntro.CurrentX = 32: frmIntro.CurrentY = 52
   ForeColor = QBColor(8)
   Print "Deposition Viewer"
   frmIntro.CurrentX = 30: frmIntro.CurrentY = 50
   ForeColor = QBColor(15)
   Print "Deposition Viewer"
   frmIntro.CurrentX = 31: frmIntro.CurrentY = 51
   ForeColor = QBColor(7)
   Print "Deposition Viewer"
   '*****************************
   FontSize = 21
   frmIntro.CurrentX = 82: frmIntro.CurrentY = 302
   ForeColor = QBColor(8)
   Print "Copyright 1996, Luke Spence"
   frmIntro.CurrentX = 80: frmIntro.CurrentY = 300
   ForeColor = QBColor(15)
   Print "Copyright 1996, Luke Spence"
   frmIntro.CurrentX = 81: frmIntro.CurrentY = 301
   ForeColor = QBColor(7)
   Print "Copyright 1996, Luke Spence"
End Sub

Private Sub exitSpalshScreen()
   frmDepoView.Show
   frmIntro.Hide
   Unload frmIntro
End Sub

Private Sub Form_Click()
   exitSpalshScreen
End Sub

Private Sub Form_KeyPress(KeyAscii As Integer)
   exitSpalshScreen
End Sub

Private Sub Form_Load()
   frmIntro.Left = (Screen.Width - frmIntro.Width) / 2
   frmIntro.Top = (Screen.Height - frmIntro.Height) / 2
   Load frmDepoView
End Sub

Private Sub Form_Paint()
   Call displayText
End Sub

Private Sub Timer1_Timer()
   exitSpalshScreen
End Sub
```

```
frmCases - 1

Option Explicit
DefInt A-Z
Dim sNewCase$

Private Sub cmdAdd_Click()
   pctAddCase.Top = lstCases.Top
   pctAddCase.Left = lstCases.Left
   lstCases.Visible = False
   pctAddCase.Visible = True
End Sub

Private Sub cmdCancelNames_Click()
   lstCases.Visible = True
   pctAddCase.Visible = False
End Sub

Private Sub cmdOK_Click()
   sNewCase$ = txtClient.Text & "/" & txtMatter.Text
   lstCases.AddItem sNewCase$
   lstCases.Visible = True
   pctAddCase.Visible = False
End Sub

Private Sub Form_Load()
   frmCases.Left = (Screen.Width - frmCases.Width) / 2
   frmCases.Top = (Screen.Height - frmCases.Height) / 2
   Dim sCaseInput$
   Dim sCaseName$
   Dim sCasePath$
   Dim sFileName$
   sFileName$ = App.Path & "cases.inf"
   sFileName$ = "C:\vb30\cases.inf"
   Open sFileName$ For Input As #1
   Do Until EOF(1)
      Line Input #1, sCaseInput$
      sCaseName$ = Mid$(sCaseInput$, 1, InStr(sCaseInput$, "*") - 1)
      lstCases.AddItem sCaseName$
   Loop
   Close #1
End Sub
```

```
frmBrowse - 1

Option Explicit
DefInt A-Z
Dim sDirectoryPath$

Private Sub cmdCancel_Click()
   Call frmBrowseQuit
End Sub

Private Sub cmdOK_Click()
   'code to send back the selected directory name
   frmDepoView.Caption = sDirectoryPath$
   Call frmBrowseQuit
End Sub

Private Sub Dir1_Change()
   sDirectoryPath$ = dir1.Path
End Sub

Private Sub Drive1_Change()
   On Error Resume Next
   dir1.Path = Drive1.Drive
End Sub

Private Sub Form_Load()
   frmBrowse.Left = (Screen.Width - frmBrowse.Width) / 2
   frmBrowse.Top = (Screen.Height - frmBrowse.Height) / 2
   dir1.Path = "\"
End Sub

Private Sub frmBrowseQuit()
   frmBrowse.Hide
   Unload frmBrowse
End Sub
```

```
frmAbout - 1

Option Explicit
DefInt A-Z

Private Sub cmdOK_Click()
   frmAbout.Hide
   Unload frmAbout
End Sub

Private Sub Form_Load()
   frmAbout.Left = (Screen.Width - frmAbout.Width) / 2
   frmAbout.Top = (Screen.Height - frmAbout.Height) / 2
   ScaleMode = 3
End Sub
```



```
Form1 - 1

' Copyright (C) 1995,1996, 1997  Tempest Software
' Last modified on 05/02/97

Option Explicit
DefInt A-Z

Dim cancelButton%
Dim totalFiles%
Dim totalProcessed%
Dim percentDone%
Dim x%
Dim numOfFiles%
Dim fileToGet$
Private Sub cmdCancel_Click()
  cancelButton% = True
  File1.Refresh
End Sub

Private Sub cmdClose_Click()
  End
End Sub

Private Sub cmdPrint_Click()
  On Error GoTo printError
  Dim startPos&
  Dim endPos&
  Dim filePath$
  Dim fileName$
  Dim allFilesToPrint$
  Dim I%
  Dim tiffPages%
  cancelButton% = False
  totalProcessed% = 0
  totalFiles% = 0
  startPos = 1
  If File1.fileName = "" Then 'If no file selected, display message & abort
    MsgBox "Please choose a file.", 0, "No file selected."
    Exit Sub
  End If
  If Right(File1.Path, 1) = "\" Then
    filePath = File1.Path
  Else
    filePath = File1.Path & "\"
  End If
  numOfFiles% = File1.ListCount ' determine number of files for percent display
  For x% = 0 To numOfFiles% - 1
    If File1.Selected(x%) Then
      totalFiles% = totalFiles% + 1
      allFilesToPrint$ = allFilesToPrint$ & File1.List(x) & " "
    End If
  Next x%
  Call buttonDisable
  Do Until startPos >= Len(allFilesToPrint$)
    endPos = InStr(startPos, allFilesToPrint$, " ")
    fileName$ = Mid$(allFilesToPrint$, startPos&, endPos& - startPos&)
    startPos = endPos + 1
    fileToGet$ = filePath$ & fileName$
    ImageMan1.Picture = fileToGet$
    tiffPages% = ImageMan1.Pages
    totalProcessed% = totalProcessed% + 1
    percentDone% = totalProcessed% / totalFiles% * 100
    lblDocsDone.Caption = "Printing document " & totalProcessed% & " of " & totalFiles% & ".  ("
& percentDone% & "%)"

    Printer.Print ""
    Printer.Print ""
    Printer.FontBold = True
    Printer.FontSize = 50
    Printer.Print fileName$
```

```
Form1 - 2

    Printer.Print tiffPages & " page(s)"
    'Printer.Line (0, 0)-(Printer.ScaleWidth, 0)
    Printer.NewPage

    For I% = 0 To tiffPages% - 1
      Printer.Print ""
      Printer.ScaleMode = vbTwips
      ImageMan1.PrnHdc = Printer.hDC
      ImageMan1.DstLeft = 0
      ImageMan1.DstTop = 0
      ImageMan1.DstRight = Printer.ScaleWidth
      ImageMan1.DstBottom = Printer.ScaleHeight
      ImageMan1.PageNumber = I%
      ImageMan1.Refresh
      percentDone% = (I% + 1) / (tiffPages% + 1) * 100
      lblPagesDone.Caption = "Printing page " & I% + 1 & " of " & tiffPages% & ".  (" & percentD
one% & "%)"
      DoEvents
      ImageMan1.PrintImage
      Printer.NewPage
      DoEvents
    Next I%
     Printer.EndDoc
    DoEvents
    If cancelButton% = True Then
      cancelButton% = False 'reset the cancel button
      Call buttonEnable
      File1.Refresh
      Printer.EndDoc
      Printer.KillDoc
      ImageMan1.Picture = ""
      'pnlPagesDone.FloodPercent = 0 ' reset the bar
      'pnlPercentDone.FloodPercent = 0 ' reset the bar
      lblDocsDone.Caption = ""
      lblPagesDone.Caption = ""
      Exit Do
    End If
  Loop
  ImageMan1.Picture = ""
  'pnlPercentDone.FloodPercent = 0
  lblDocsDone.Caption = ""
  lblPagesDone.Caption = ""
  Call buttonEnable
  File1.Refresh
  Exit Sub

printError:
  MsgBox "Error # " & Str(Err.Number) & " was generated." & Chr(13) & Err.Description, , "Error"
, Err.HelpFile, Err.HelpContext
  Resume Next
End Sub


Private Sub Dir1_Change()
  File1.Path = Dir1.Path
End Sub

Private Sub Drive1_Change()
  On Error Resume Next
  Dir1.Path = Drive1.Drive
End Sub


Private Sub Form_Load()
  Me.Left = (Screen.Width - Me.Width) / 2
  Me.Top = (Screen.Height - Me.Height) / 3
  On Error Resume Next
  cancelButton% = False
End Sub
```

Form1 - 3

```
Public Sub buttonDisable()
  File1.Enabled = False
  Dir1.Enabled = False
  Drive1.Enabled = False
  cmdPrint.Enabled = False
  cmdClose.Enabled = False
End Sub

Public Sub buttonEnable()
  File1.Enabled = True
  Dir1.Enabled = True
  Drive1.Enabled = True
  cmdPrint.Enabled = True
  cmdClose.Enabled = True
End Sub
```

Paper Chaser Table Structure

Court Information
Disk Names
Document Info
List of Privileges
Preferences
Report Names

Court Information

Field Name	Data Type	Length
Cause #	Text	250
Plaintiff	Text	250
Defendant	Text	250
Court	Text	250
County	Text	250
State	Text	250
Title of Pleading	Text	250

Disk Names

Field Name	Data Type	Length
DiskName	Text	50
DOCN1	Text	50
DOCN2	Text	50

Document Info

Field Name	Data Type	Length
Doc Number	Text	9
Entry Info	Text	12
Doc Date	Date/Time	
Beg Bates #	Text	20
End Bates #	Text	20
To	Text	250
From	Text	250
CC's	Text	250
Description	Memo	64,000
Marked	Text	2
Offered	Text	2
Admitted	Text	2
Date	Text	20
Tag 1	Text	2
Tag 2	Text	2
Tag 3	Text	2
Tag 4	Text	2
Tag 5	Text	2
Tag 6	Text	2
Tag 7	Text	2
Tag 8	Text	2
Tag 9	Text	2
Tag 10	Text	2
Tag 11	Text	2
Tag 12	Text	2
Tag 13	Text	2
Tag 14	Text	2
Tag 15	Text	2
Tag 16	Text	2
Tag 17	Text	2
Tag 18	Text	2
Tag 19	Text	2
Tag 20	Text	2
Tag 21	Text	2
Tag 22	Text	2
Tag 23	Text	2
Tag 24	Text	2
Produced	Text	1
Comments	Memo	64,000
Privilege Log	Text	25
Revised By	Text	3
Date Revised	Date/Time	
Print	Text	20
Exhibit #	Number	Long Integer (-2,147,483,648 to 2,147,483,647)
Ext	Text	6
Order of Display	Number	Long Integer (-2,147,483,648 to 2,147,483,647)
Disposition	Text	20

List of Privileges

Field Name	Data Type	Length
Privileges	Text	25

Preferences

Field Name	Data Type	Length
Case Name		
View Screen Message		
Field 1	Text	50
Field 2	Text	50
Field 3	Text	50
Field 4	Text	50
Field 5	Text	50
Field 6	Text	50
Field 7	Text	50
Field 8	Text	50
Field 9	Text	50
Field 10	Text	50
Field 11	Text	50
Field 12	Text	50
Field 13	Text	50
Field 14	Text	50
Field 15	Text	50
Field 16	Text	50
Field 17	Text	50
Field 18	Text	50
Field 19	Text	50
Field 20	Text	50
Field 21	Text	50
Field 22	Text	50
Field 23	Text	50
Field 24	Text	50
ImagePath	Text	100
IsysPath	Text	100
DbPath1	Text	100

Report Names

Field Name	Data Type	Length
Report 1	Text	50
Report 2	Text	50
Report 3	Text	50
Report 4	Text	50
Report 5	Text	50
Report 6	Text	50
Report 7	Text	50
Report 8	Text	50
Report 9	Text	50
Report 10	Text	50
Report 11	Text	50
Report 12	Text	50
Report 13	Text	50
Report 14	Text	50
Report 15	Text	50
Report 16	Text	50
Report 17	Text	50
Report 18	Text	50
Report 19	Text	50
Report 20	Text	50
Report 21	Text	50
Report 22	Text	50
Report 24	Text	50
Report 24	Text	50

```
frmAutomate = 1

'Automated Document Separator
'Copyright (c) 1996, Luke Spence
Option Explicit
DefInt A-Z
Private Declare Function GetPixel Lib "GDI" (ByVal hDC As Integer, ByVal x As Integer, ByVal Y A
s Integer) As Long
Dim lColor&
Dim iX%
Dim iY%
Dim sFilename$
Dim iPageCount%
Dim iCurrentPage%
Dim iNewFilesFilename$
Dim iNewFilePageCount%
Dim iNewFileCurrentPage%
Dim sValidDivider$
Dim iFileNumber%
Dim sNewFilename$

Private Sub cmdFileOpen_Click()
   On Error GoTo errorLoad
   Dim iDocCount%
   iDocCount% = 0
   If txtDocNameNumber.Text = "" Then Exit Sub
   iFileNumber% = frmAutomate.txtDocNameNumber.Text
   iCurrentPage% = 0
   frmAutomate.cdlFilename.filename = "*.tif"
   frmAutomate.cdlFilename.InitDir = "c:\pictures\demo\"
   frmAutomate.cdlFilename.Action = 1
   On Error Resume Next
   If frmAutomate.cdlFilename = "*.tif" Then Exit Sub
   File1.Path = "c:\pictures\demo\"
   File1.Visible = True
   frmAutomate.TIFF.File = frmAutomate.cdlFilename.filename
   Call pagesTotal
   frmAutomate.lblNumOfPages.Caption = "Number Of Pages: " & iPageCount%
   Do While iCurrentPage% < iPageCount%
      iCurrentPage% = iCurrentPage% + 1
      frmAutomate.lblCurrentPage.Caption = "Current Page: " & iCurrentPage%
      Call imageDisplay
      Call examinePage
      File1.Refresh
      DoEvents
      If sValidDivider$ = "T" Then
         iFileNumber% = iFileNumber% + 1
         iDocCount% = iDocCount% + 1
      Else
         Call writeTIF
      End If
   Loop
   iDocCount% = iDocCount% + 1
   'reset everything back to original status
   lblCurrentPage.Caption = ""
   lblNumOfPages.Caption = ""
   txtDocNameNumber.Text = ""
   txtDocNamePrefix.Text = ""
   cmdFileOpen.Enabled = False
   frmAutomate.TIFF.File = ""
   frmAutomate.TIFF.Repaint = True
   MsgBox "  " & iPageCount% & " pages were separated into " & iDocCount% & " documents.", 0, "D
ocument Separtion Complete."
   Exit Sub
errorLoad:
   'reset everything back to original status
   lblCurrentPage.Caption = ""
   lblNumOfPages.Caption = ""
   txtDocNameNumber.Text = ""
   txtDocNamePrefix.Text = ""
   cmdFileOpen.Enabled = False
```

```
frmAutomate - 2

   frmAutomate.TIFF.File = ""
   frmAutomate.TIFF.Repaint = True
   Exit Sub
End Sub

Private Sub examinePage()
   Dim sColorCount$
   Dim iQuadrant%
   Dim iWidthSection%
   Dim iHeightSection%
   Dim iPixelCheck%
   Dim iBlackCount%
   Dim iWhiteCount%
   Dim iLargeX%
   Dim iLargeY%
   Dim iSmallX%
   Dim iSmallY%
   iWidthSection% = frmAutomate.TIFF.BitmapWidth / 8
   iHeightSection% = frmAutomate.TIFF.BitmapHeight / 8
   frmAutomate.TIFF.Visible = True
   frmAutomate.TIFF.BitmapDC = True
   For iQuadrant% = 1 To 4
      Select Case iQuadrant%
      Case 1
         iSmallX% = iWidthSection% * 1
         iSmallY% = iHeightSection% * 1
         iLargeX% = iWidthSection% * 3
         iLargeY% = iHeightSection% * 3
      Case 2
         iSmallX% = iWidthSection% * 5
         iSmallY% = iHeightSection% * 1
         iLargeX% = iWidthSection% * 7
         iLargeY% = iHeightSection% * 3
      Case 3
         iSmallX% = iWidthSection% * 1
         iSmallY% = iHeightSection% * 5
         iLargeX% = iWidthSection% * 3
         iLargeY% = iHeightSection% * 7
      Case 4
         iSmallX% = iWidthSection% * 5
         iSmallY% = iHeightSection% * 5
         iLargeX% = iWidthSection% * 7
         iLargeY% = iHeightSection% * 7
      End Select
      For iPixelCheck% = 1 To 1000
         iX% = Int((iLargeX% - iSmallX% + 1) * Rnd + iSmallX%)
         iY% = Int((iLargeY% - iSmallY% + 1) * Rnd + iSmallY%)
         lColor& = GetPixel(TIFF.BitmapDC, iX%, iY%)
         If lColor& = 0 Then
            iBlackCount% = iBlackCount% + 1
         Else
            iWhiteCount% = iWhiteCount% + 1
         End If
      Next iPixelCheck%
      Select Case iQuadrant%
      Case 1, 4
         If iWhiteCount% > 100 Then
            frmAutomate.TIFF.BitmapDC = False
            sValidDivider$ = "F"
            Exit Sub
         End If
      Case 2, 3
         If iBlackCount% > 100 Then
            frmAutomate.TIFF.BitmapDC = False
            sValidDivider$ = "F"
            Exit Sub
         End If
      End Select
   iBlackCount% = 0
```

```
frmAutomate - 3

   iWhiteCount% = 0
   Next iQuadrant%
   svalidDividers = "T"
   frmAutomate.TIFF.BitmapDC = False
End Sub

Private Sub Form_Load()
   frmAutomate.ScaleWidth = 600
   frmAutomate.ScaleHeight = 450
   frmAutomate.Top = (Screen.Height - frmAutomate.Height) / 2
   frmAutomate.Left = (Screen.Width - frmAutomate.Width) / 2
   lblCurrentPage.Caption = ""
   lblNumOfPages.Caption = ""
   Randomize
End Sub

Private Sub imageDisplay()
   frmAutomate.TIFF.FilePage = iCurrentPage%
   frmAutomate.TIFF.File = frmAutomate.cdlFilename.filename
   frmAutomate.TIFF.ImageWidth = frmAutomate.TIFF.Width
   frmAutomate.TIFF.ImageHeight = frmAutomate.TIFF.Height
   frmAutomate.TIFF.Repaint = True
End Sub

Private Sub pagesTotal()
   frmAutomate.TIFF.InfoPage = 32767
   frmAutomate.TIFF.InfoFile = frmAutomate.cdlFilename.filename
   iPageCount% = TIFF.InfoPage
End Sub

Private Sub txtDocNameNumber_Change()
   cmdFileOpen.Enabled = True
End Sub

Private Sub writeTIF()
   frmAutomate.TIFF.FilePage = iCurrentPage%
   frmAutomate.TIFF.SaveMulti = True
   frmAutomate.TIFF.SaveFormat = LVB_FILE_CCITT_GROUP4
   frmAutomate.TIFF.SaveFile = txtDocNamePrefix.Text & "-" & iFileNumber% & ".tif"
End Sub
```

```
frmAutomate - 1

VERSION 5.00
Begin VB.Form frmAutomate
   Appearance      =   0  'Flat
   AutoRedraw      =   -1  'True
   BackColor       =   &H008080000&
   Caption         =   "Automated Document Separator"
   ClientHeight    =   6105
   ClientLeft      =   1800
   ClientTop       =   720
   ClientWidth     =   6225
   BeginProperty Font
      Name            =   "MS Sans Serif"
      Size            =   8.25
      Charset         =   0
      Weight          =   700
      Underline       =   0   'False
      Italic          =   0   'False
      Strikethrough   =   0   'False
   EndProperty
   ForeColor       =   &H80000008&
   Icon            =   (Icon)
   LinkTopic       =   "Form1"
   MaxButton       =   0   'False
   PaletteMode     =   1  'UseZOrder
   ScaleHeight     =   407
   ScaleMode       =   3  'Pixel
   ScaleWidth      =   415
   Begin VB.FileListBox File1
      Appearance      =   0  'Flat
      BackColor       =   &H008080000&
      ForeColor       =   &H00FFFFFF&
      Height          =   5880
      Left            =   4560
      TabIndex        =   8
      Top             =   120
      Visible         =   0   'False
      Width           =   1455
   End
   Begin VB.TextBox txtDocNamePrefix
      Appearance      =   0  'Flat
      BackColor       =   &H008080000&
      Height          =   285
      Left            =   1800
      TabIndex        =   5
      Top             =   5400
      Width           =   615
   End
   Begin VB.TextBox txtDocNameNumber
      Appearance      =   0  'Flat
      BackColor       =   &H008080000&
      Height          =   285
      Left            =   1800
      TabIndex        =   4
      Top             =   5760
      Width           =   615
   End
   Begin VB.PictureBox cdlFilename
      Appearance      =   0  'Flat
      BackColor       =   &H800000005&
      ForeColor       =   &H80000008&
      Height          =   480
      Left            =   0
      ScaleHeight     =   450
      ScaleWidth      =   1170
      TabIndex        =   9
      Top             =   2400
      Width           =   1200
   End
   Begin VB.PictureBox TIFF
```

```
frmAutomate - 2
      Appearance      =   0   'Flat
      BackColor       =   &H00C0C0C0&
      ForeColor       =   &H80000008&
      Height          =   4575
      Left            =   240
      ScaleHeight     =   4545
      ScaleWidth      =   4065
      TabIndex        =   1
      Top             =   600
      Width           =   4095
   End
   Begin VB.CommandButton cmdFileOpen
      Appearance      =   0   'Flat
      Caption         =   "Open File"
      Enabled         =   0   'False
      Height          =   375
      Left            =   2640
      TabIndex        =   0
      Top             =   5520
      Width           =   1095
   End
   Begin VB.Label lblDocNameNumber
      Appearance      =   0   'Flat
      BackColor       =   &H00808000&
      Caption         =   "Start Number:"
      ForeColor       =   &H80000008&
      Height          =   255
      Left            =   480
      TabIndex        =   7
      Top             =   5760
      Width           =   1215
   End
   Begin VB.Label lblDocNamePrefix
      Appearance      =   0   'Flat
      BackColor       =   &H00808000&
      Caption         =   "Prefix:"
      ForeColor       =   &H80000008&
      Height          =   255
      Left            =   1080
      TabIndex        =   6
      Top             =   5400
      Width           =   615
   End
   Begin VB.Label lblCurrentPage
      Appearance      =   0   'Flat
      BackColor       =   &H00808000&
      ForeColor       =   &H80000008&
      Height          =   255
      Left            =   2400
      TabIndex        =   3
      Top             =   120
      Width           =   2175
   End
   Begin VB.Label lblNumOfPages
      Appearance      =   0   'Flat
      BackColor       =   &H00808000&
      ForeColor       =   &H80000008&
      Height          =   255
      Left            =   120
      TabIndex        =   2
      Top             =   120
      Width           =   2175
   End
End
```

CLAIMS

WHAT IS CLAIMED IS:

1. A method for interpreting a computer file including a plurality of pages, said method comprising:

(a) selecting a portion of a page;

(b) comparing said portion with a document separator template; and

(c) identifying a predefined image in said computer file.

2. The method of claim 1 wherein each page includes a plurality of pixel values and step (a) includes selecting a subset of the pixel values of a page.

3. The method of claim 2 wherein step (b) includes comparing the pixel values of said subset with corresponding pixel values in said document separator template.

4. The method of claim 3 wherein said document separator template includes a first area pixel of values and a second area of pixel values, wherein the first area pixel values comprise a different value than the second area pixel values.

5. The method of claim 4 wherein step (c) includes determining whether the pixel values of said subset are substantially similar to the corresponding pixel values of said document separator template.

6. The method of claim 5 further identifying a page in said computer file as a document separator page when said pixel values of said subset are substantially similar to the corresponding pixel values of said document separator template.

7. A computer readable storage medium for storing an executable set of software instructions which, when inserted into a host computer system, is capable of controlling the operation of the host computer, said software instructions being operable to identify the existence of a predefined image in a computer file including a plurality of pages, said software instructions including:

means for selecting a portion of a page;

means for comparing said portion of a page with a document separator template; and

means for identifying a predefined image in said computer file.

8. The invention of claim 7 wherein each page includes a plurality of pixel values and said means for selecting a portion of a page includes a means for selecting a subset of the pixel values of said page.

9. The invention of claim 8 wherein said means for comparing said portion of a page with a document separator template includes a means for comparing the pixel values of said subset with

corresponding pixel values in said document separator template.

10. The invention of claim 9 wherein said document separator template includes a first area pixel values and a second area of pixel values, wherein the first area pixel values comprise a different value than the second area pixel values.

11. The invention of claim 10 wherein said means for identifying a predefined image includes a means for determining whether the pixel values of said subset are substantially similar to the corresponding pixel values of said document separator template.

12. The invention of claim 11 wherein said means for identifying a page in said computer file as a document separator page includes a means for identifying a page as a document separator page when said pixel values of said subset are substantially similar to the corresponding pixel values of said document separator template.

13. A user-configurable document management system, comprising:

a computer including a display and an input device;

a scanner coupled to said computer; and

a printer coupled to said computer;

wherein said computer includes document management software including:

an image viewer;

a search engine;

a briefing tool;

a transcript viewer; and

an installation module permitting a user to configure said software to include any desired image viewer, search engine, briefing tool, and transcript viewer.

14. The document management system of claim 13 wherein said briefing tool is adapted to generate a window on said display that remains viewable in its entirety until terminated by user intervention.

15. The document management system of claim 13 wherein said transcript viewer interprets transcript files in a variety of formats and automatically determines which format is associated each transcript file.

16. The document management system of claim 15 wherein said transcript viewer further adapts each transcript file to be processed by said document management software.

17. A method for installing software comprising a plurality of modules in a computer system, comprising:

(a) recognizing the existence of installable modules;

(b) installing said installable modules.

18. A method for generating a document database, comprising:

(a) reading a look-up table;

(b) scanning a document; and

(c) storing said document in said document database at a location determined by said look-up table.

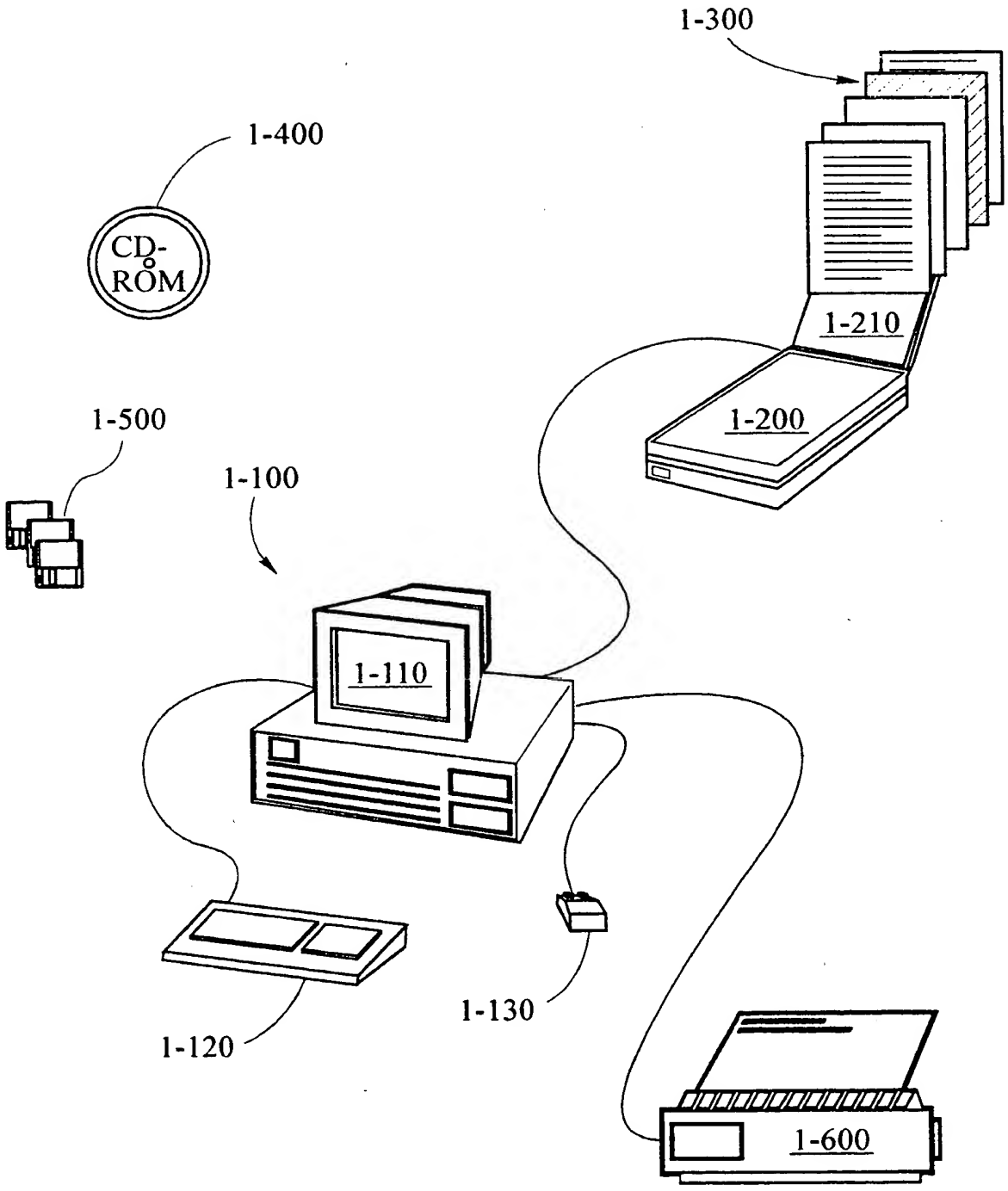


FIG 1

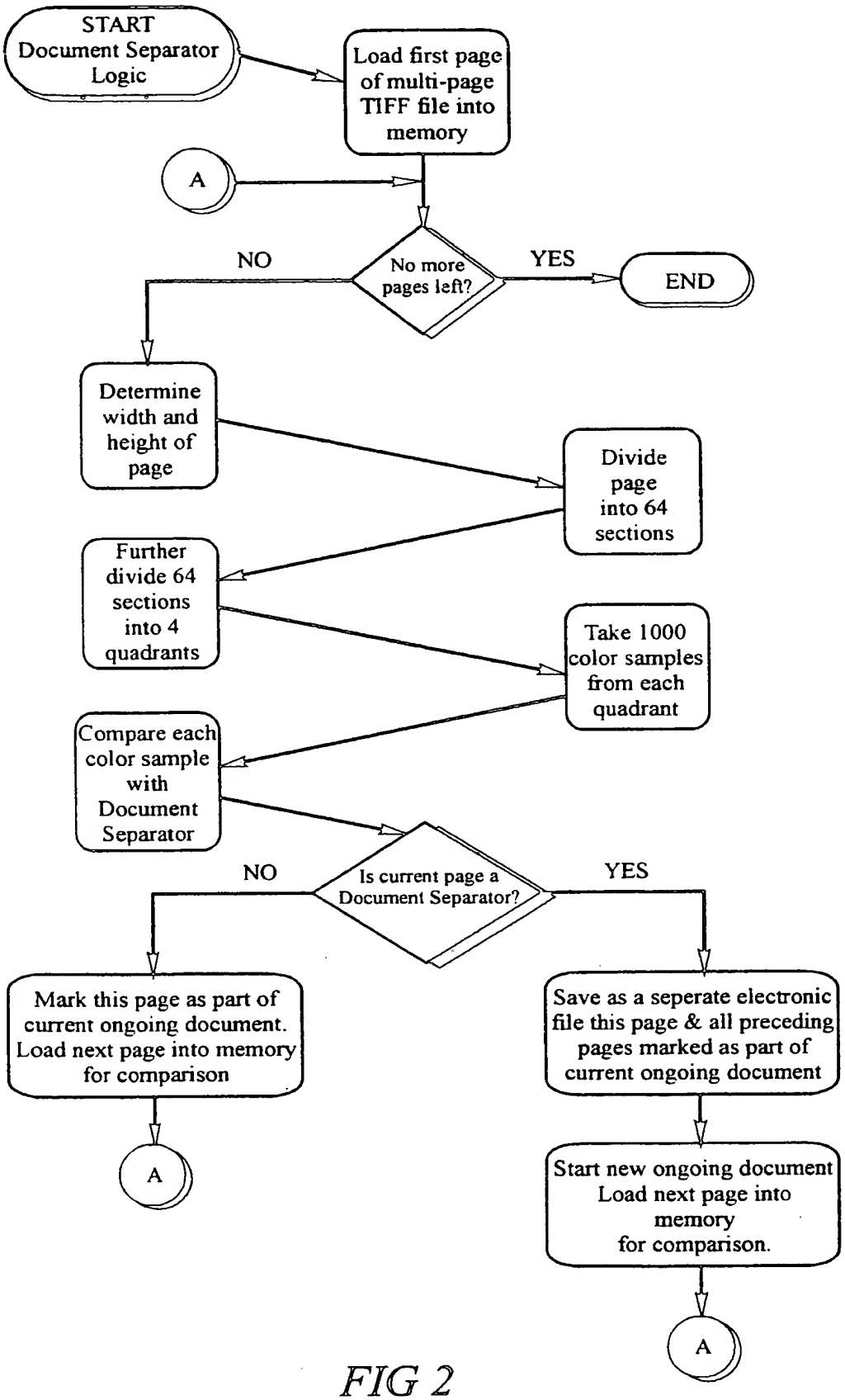


FIG 2

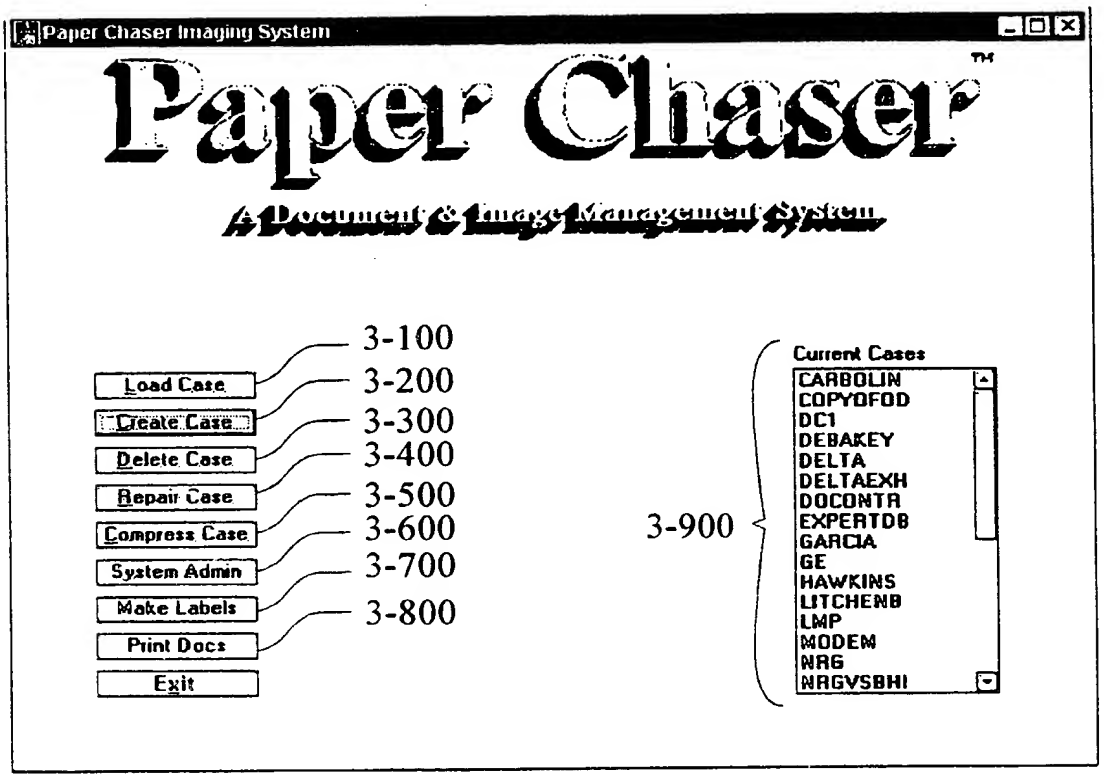


FIG 3

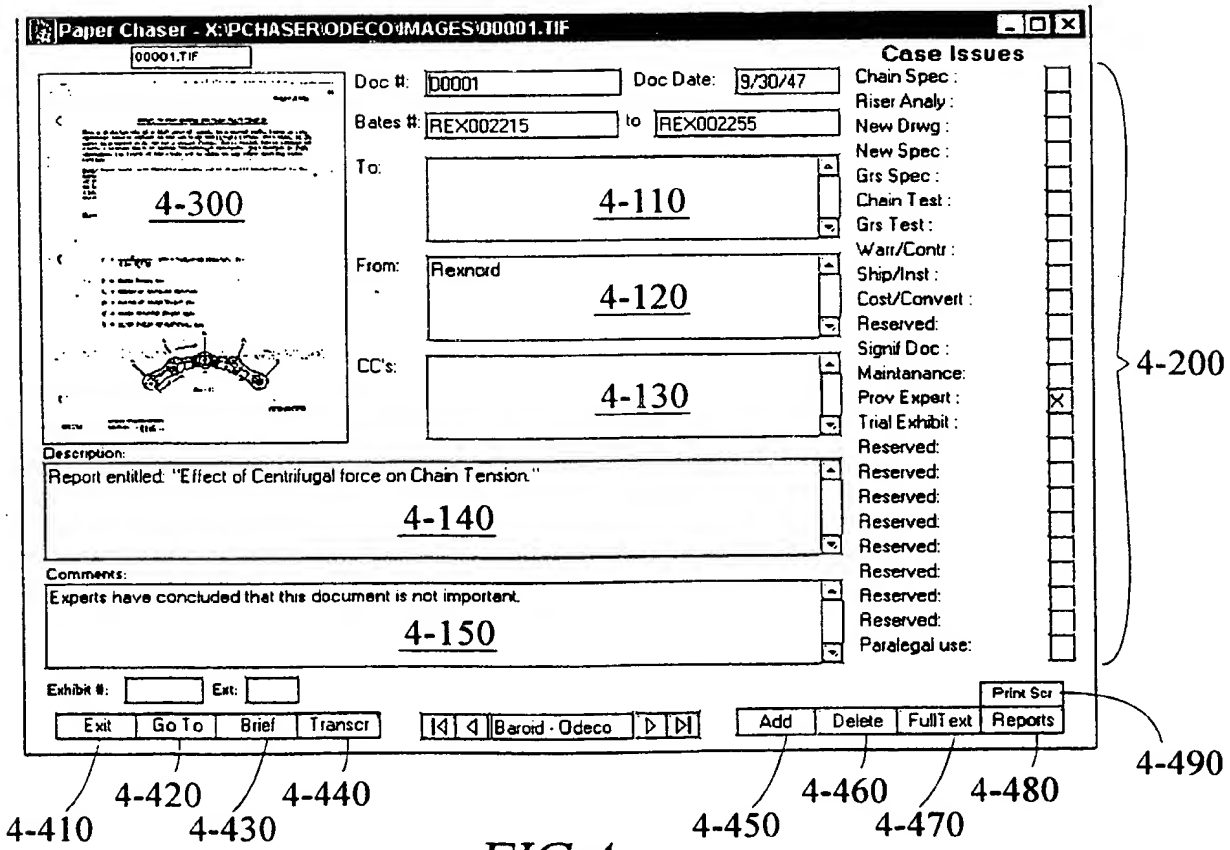


FIG 4

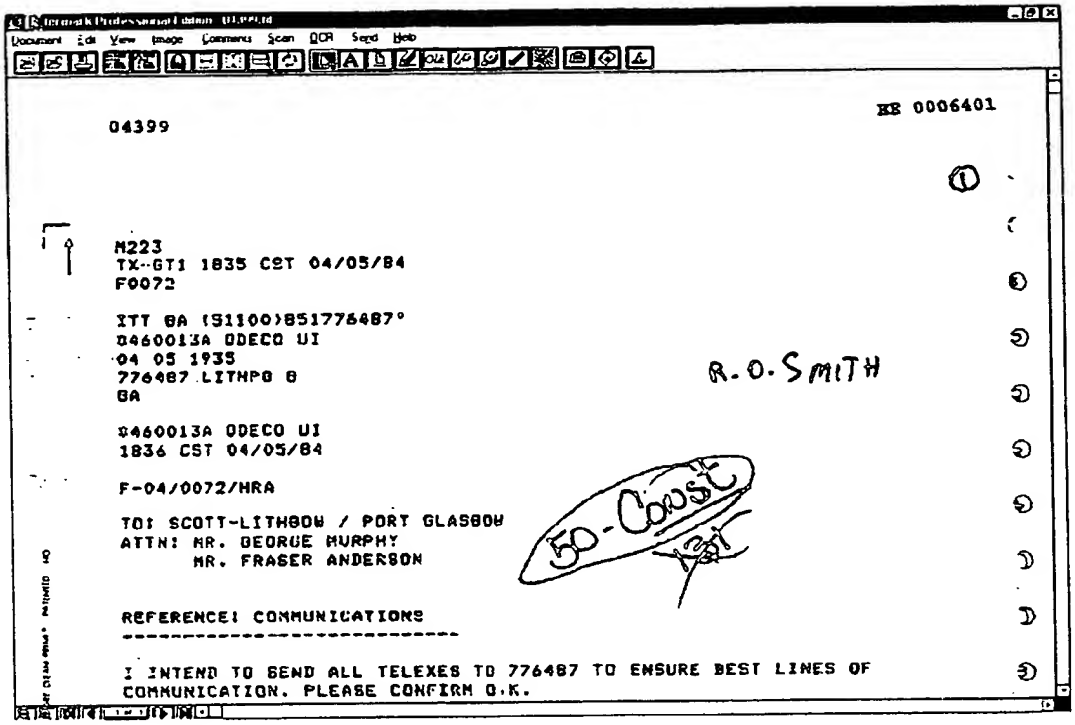


FIG 5

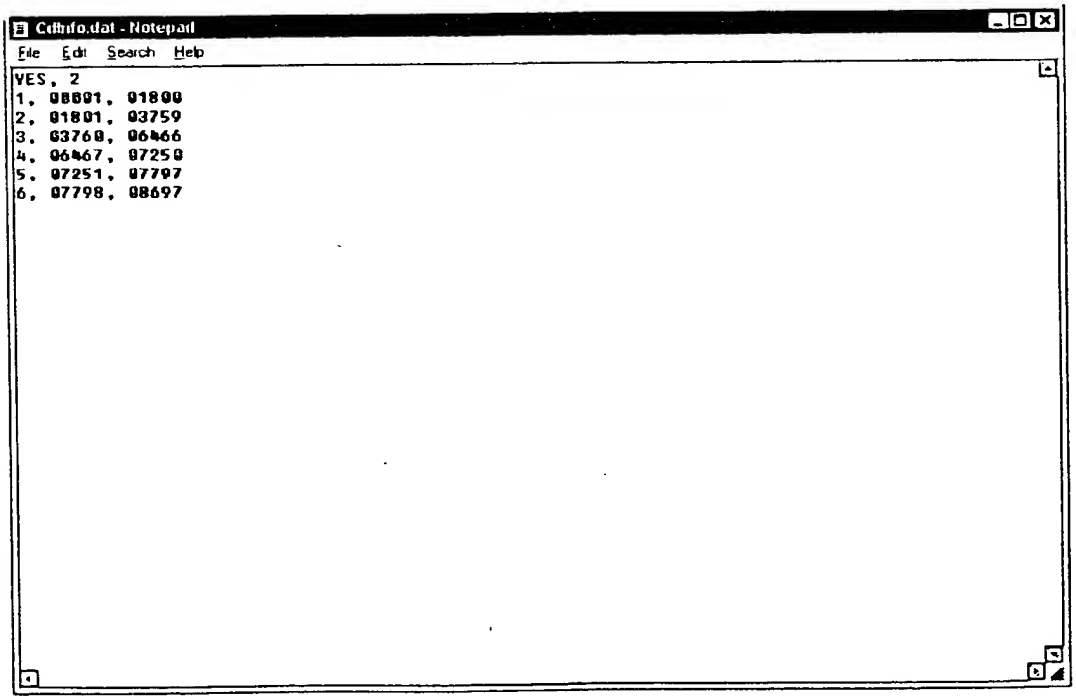
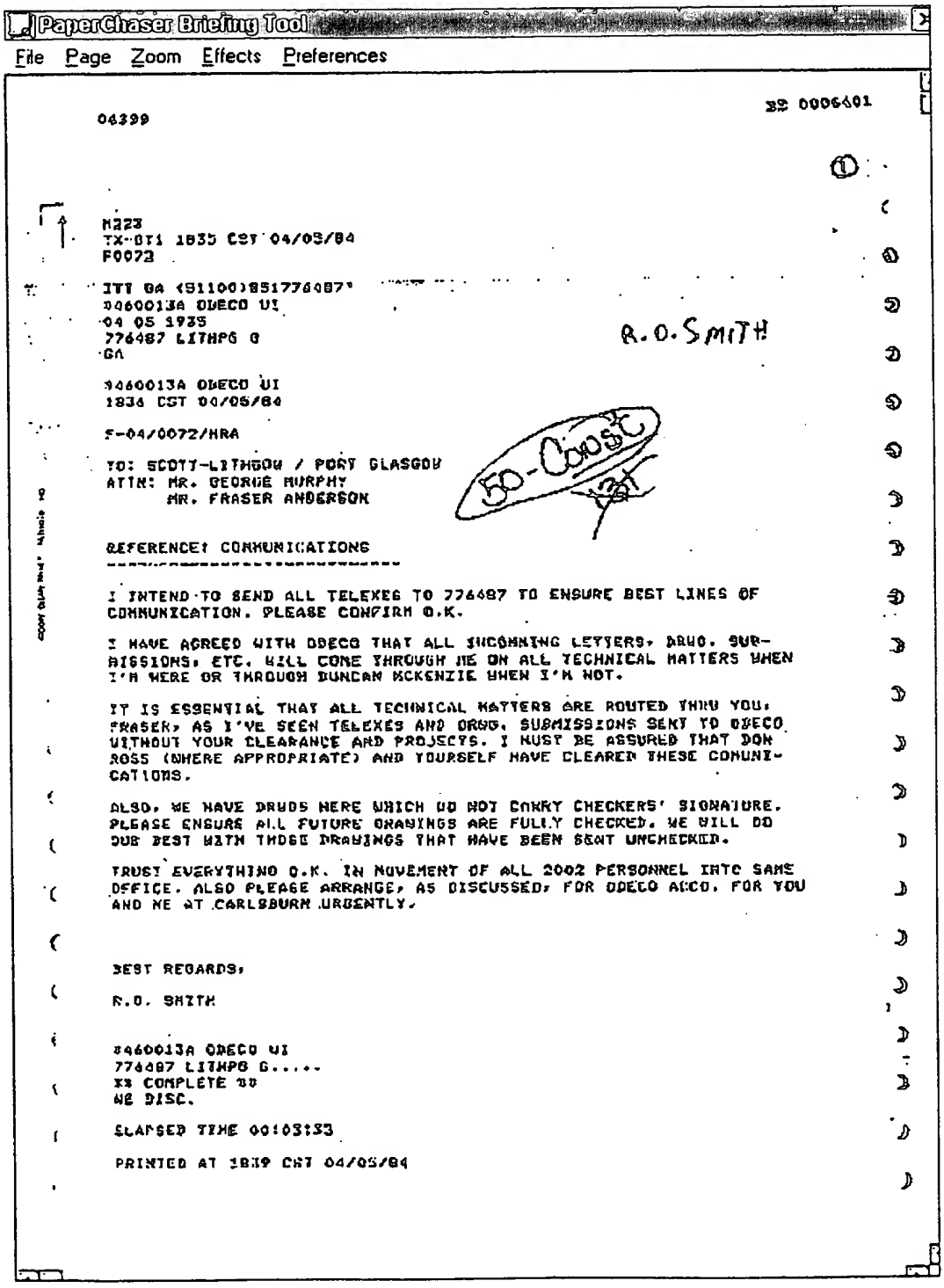


FIG 6


PaperChaser Briefing Tool

File Page Zoom Effects Preferences

04399

22 0006401

M223
TX-DT1 1835 CST 04/05/84
F0072

ITT GA (S1100)8517764877
#460013A ODECO UI
04 05 1935
776487 LITHPG G
GA

#460013A ODECO UI
1836 CST 04/05/84

F-04/0072/HRA

TO: SCOTT-LITHGOW / PORT GLASGOW
ATTN: MR. GEORGE MURPHY
MR. FRASER ANDERSON

R.O. SMITH

50-CROSS
131

REFERENCE: COMMUNICATIONS

I INTEND TO SEND ALL TELEXES TO 776487 TO ENSURE BEST LINES OF COMMUNICATION. PLEASE CONFIRM O.K.

I HAVE AGREED WITH ODECO THAT ALL INCOMMING LETTERS, DRWG. SUBMISSIONS, ETC. WILL COME THROUGH ME ON ALL TECHNICAL MATTERS WHEN I'M HERE OR THROUGH DUNCAN MCKENZIE WHEN I'M NOT.

IT IS ESSENTIAL THAT ALL TECHNICAL MATTERS ARE ROUTED THRU YOU, FRASER, AS I'VE SEEN TELEXES AND DRWG. SUBMISSIONS SENT TO ODECO WITHOUT YOUR CLEARANCE AND PROJECTS. I MUST BE ASSURED THAT DON ROSS (WHERE APPROPRIATE) AND YOURSELF HAVE CLEARED THESE COMUNICATIONS.

ALSO, WE HAVE DRWGS HERE WHICH DO NOT CARRY CHECKERS' SIGNATURE. PLEASE ENSURE ALL FUTURE DRAWINGS ARE FULLY CHECKED. WE WILL DO OUR BEST WITH THOSE DRAWINGS THAT HAVE BEEN SENT UNCHECKED.

TRUST EVERYTHING O.K. IN MOVEMENT OF ALL 2002 PERSONNEL INTO SAME OFFICE. ALSO PLEASE ARRANGE, AS DISCUSSED, FOR ODECO ACCO. FOR YOU AND ME AT CARLSBURN URGENTLY.

BEST REGARDS,

R.O. SMITH

#460013A ODECO UI
776487 LITHPG G.....
XX COMPLETE 00
NE DISC.

ELAPSED TIME 00:03:33

PRINTED AT 1839 CST 04/05/84

FIG 7

SUBSTITUTE SHEET (RULE 26)

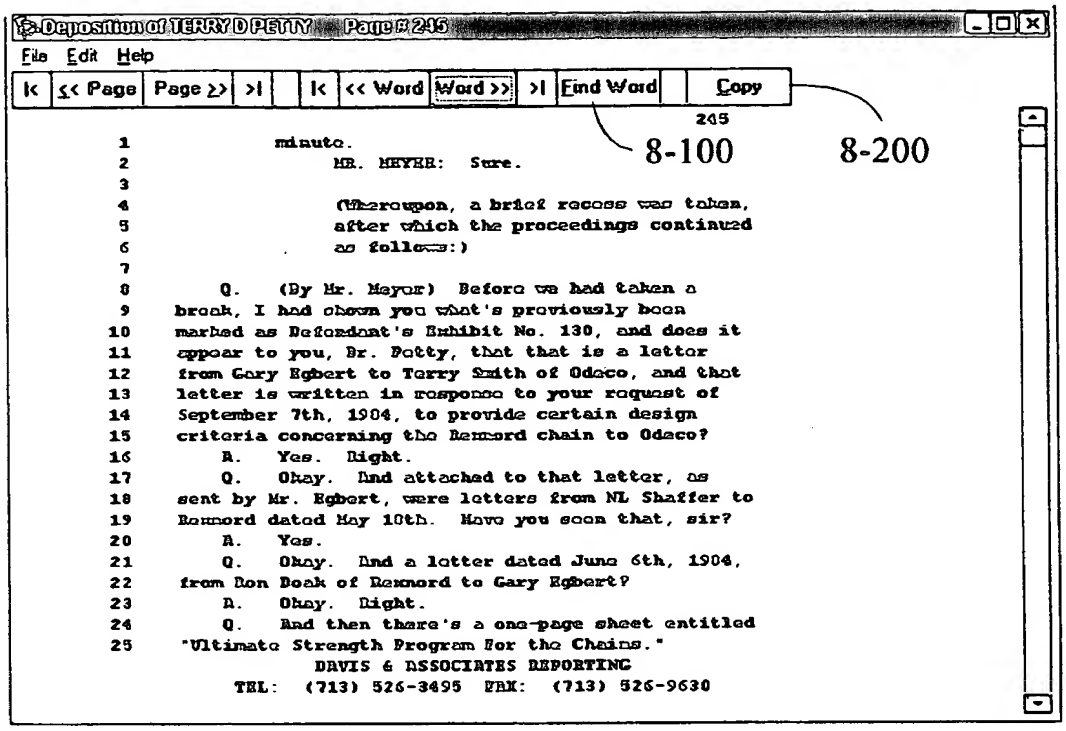


FIG 8

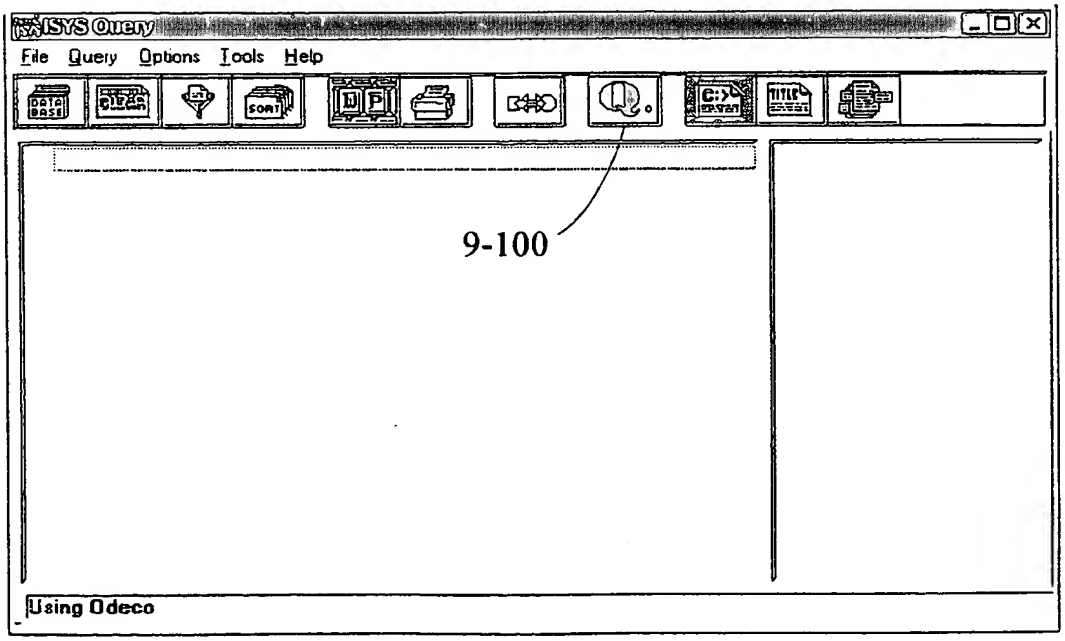


FIG 9

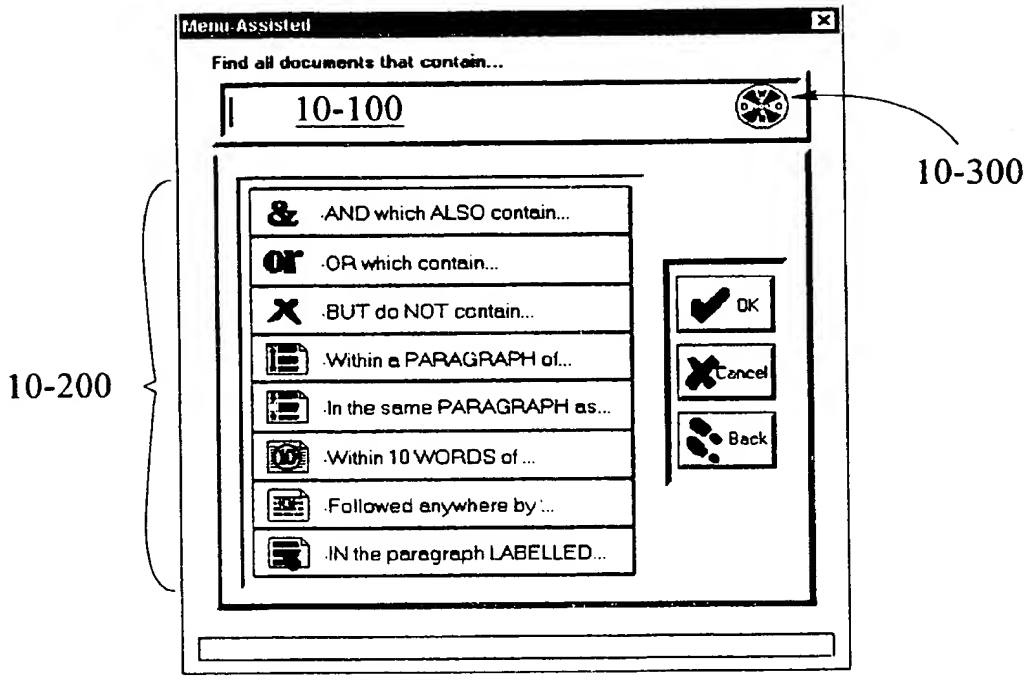


FIG 10

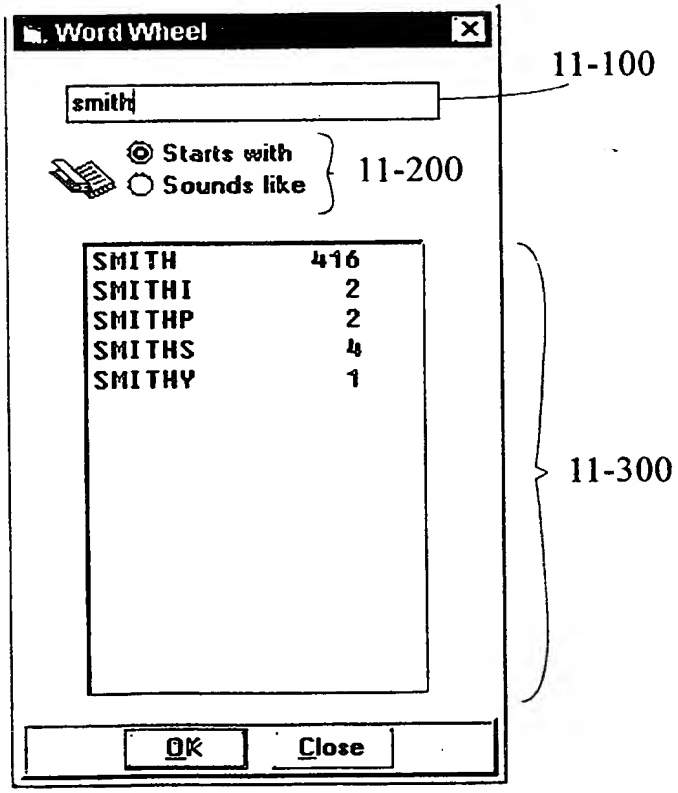


FIG 11

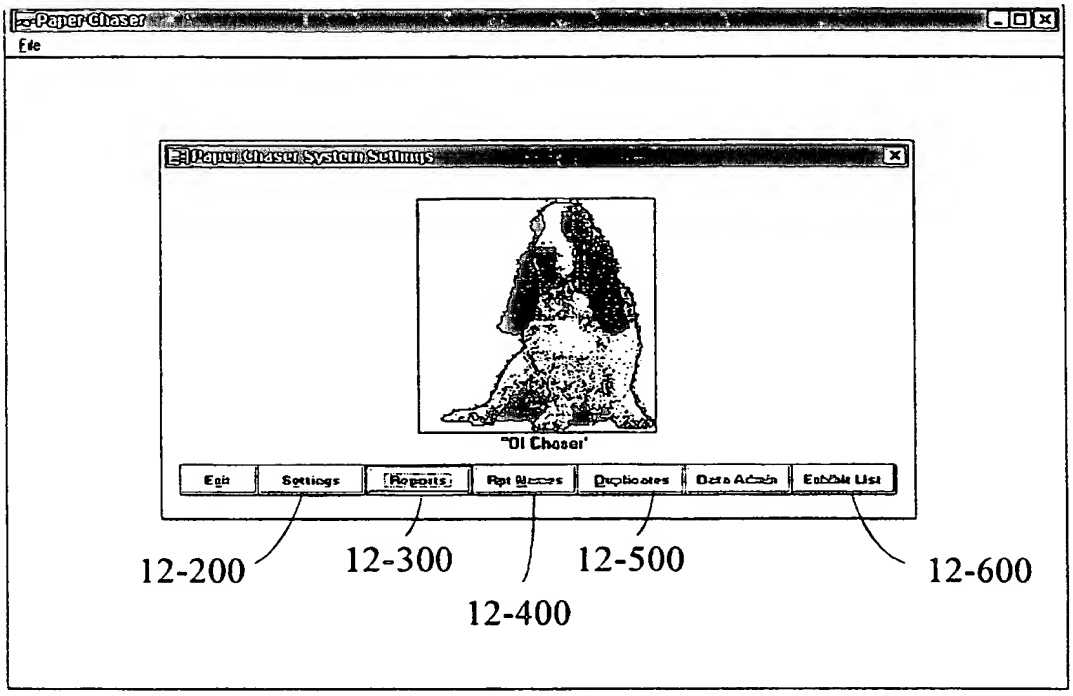


FIG 12

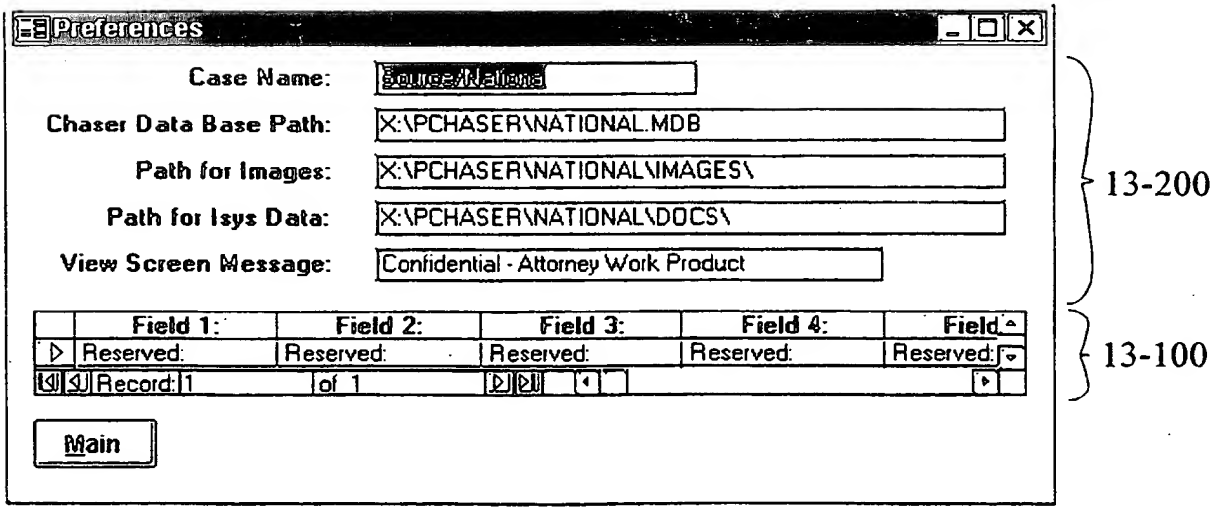


FIG 13

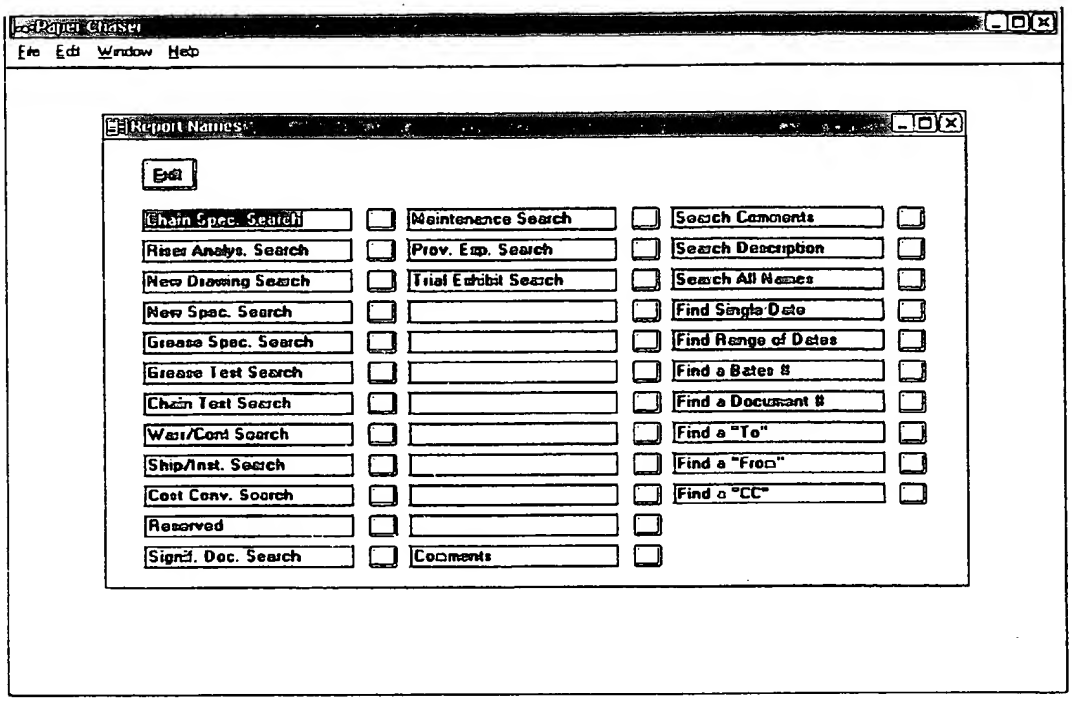


FIG 14

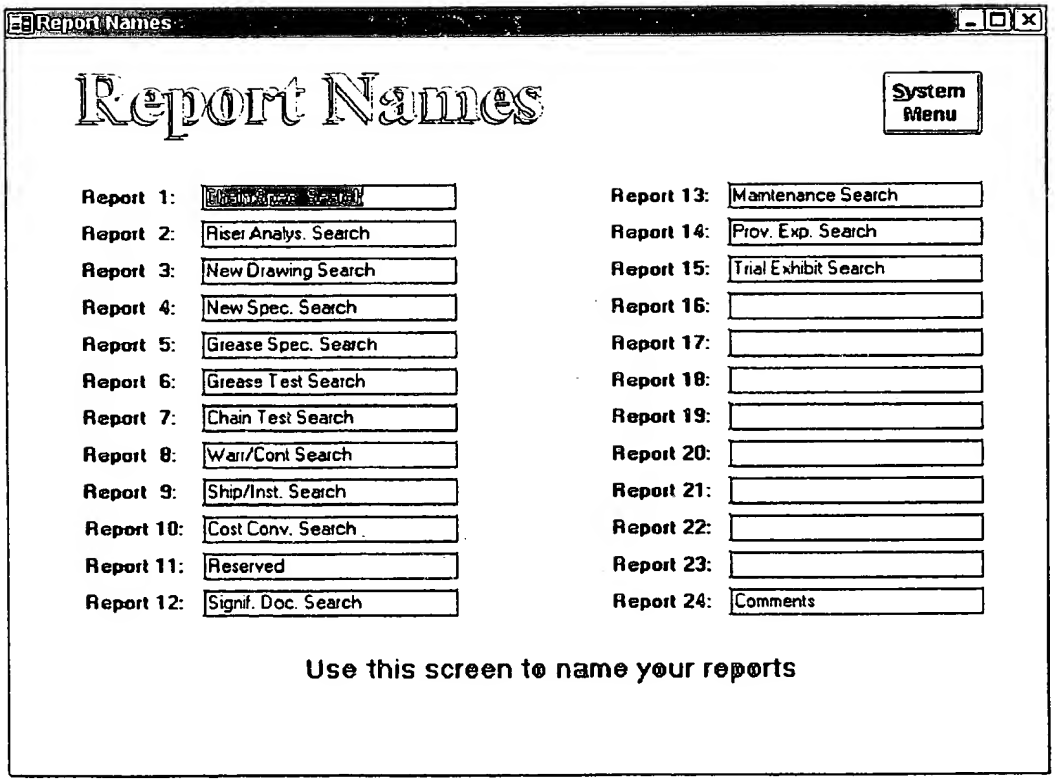


FIG 15

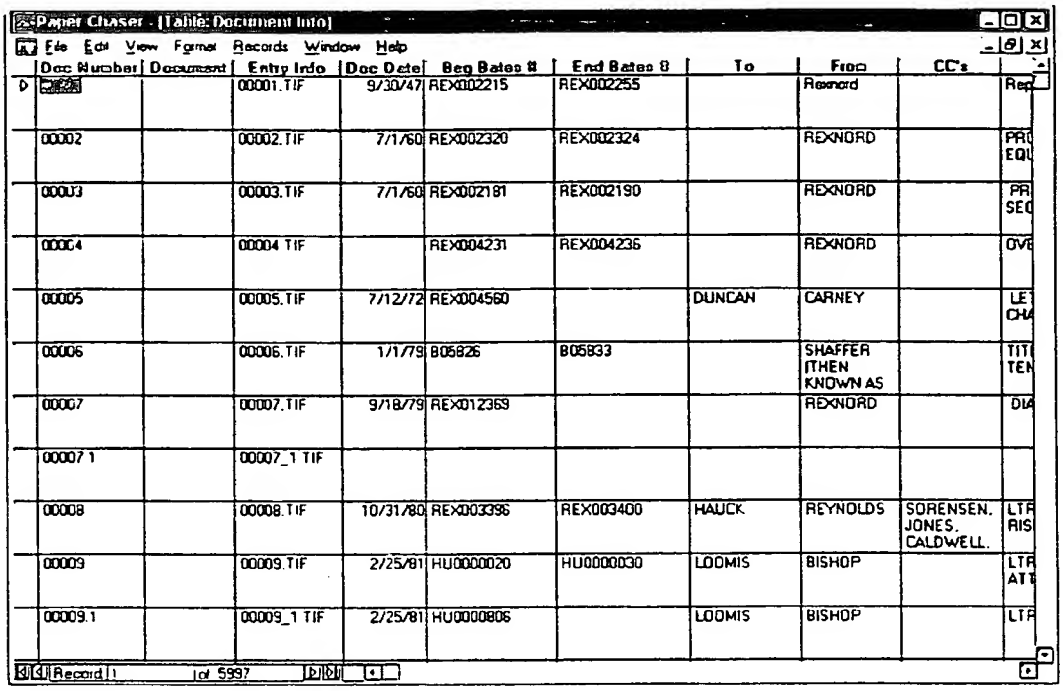


FIG 16

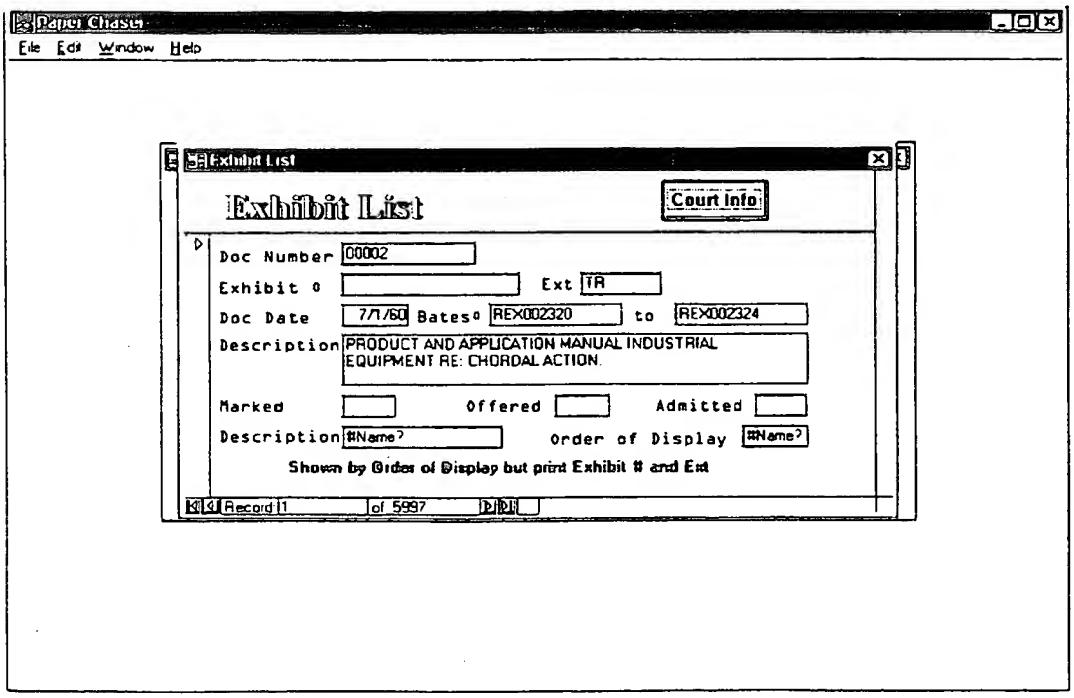


FIG 17

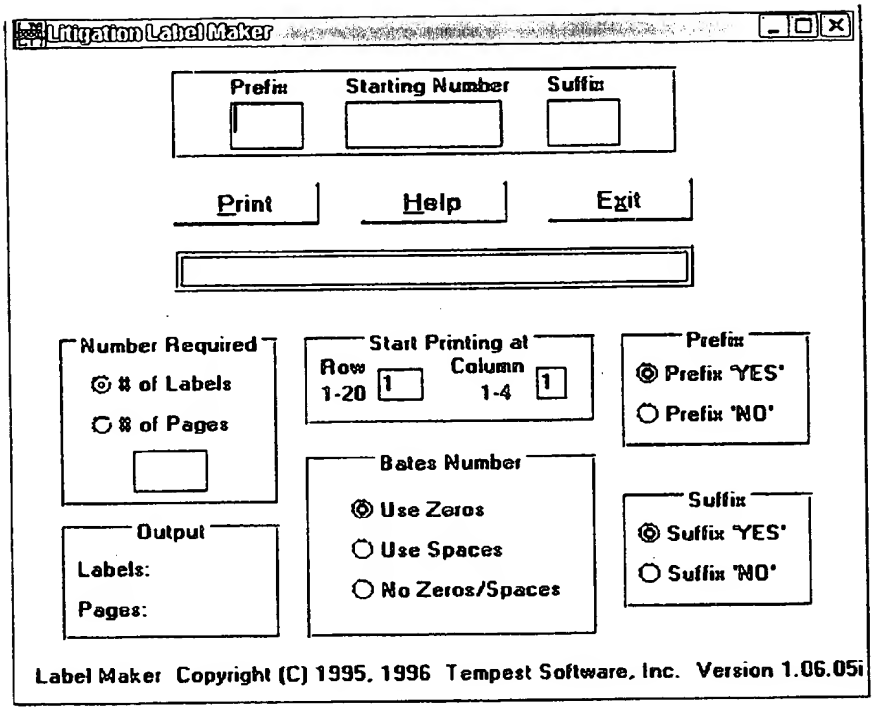


FIG 18

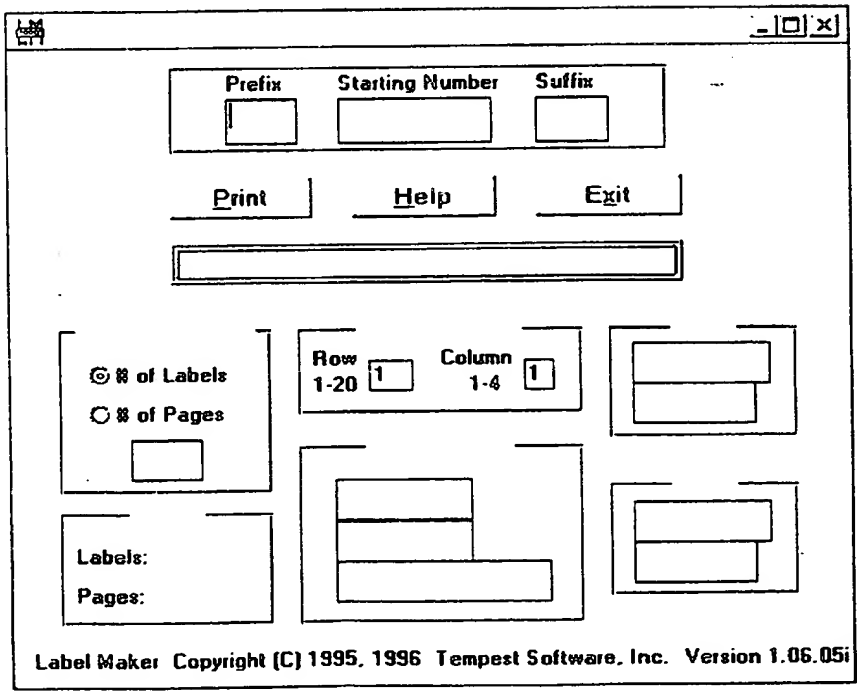


FIG 19

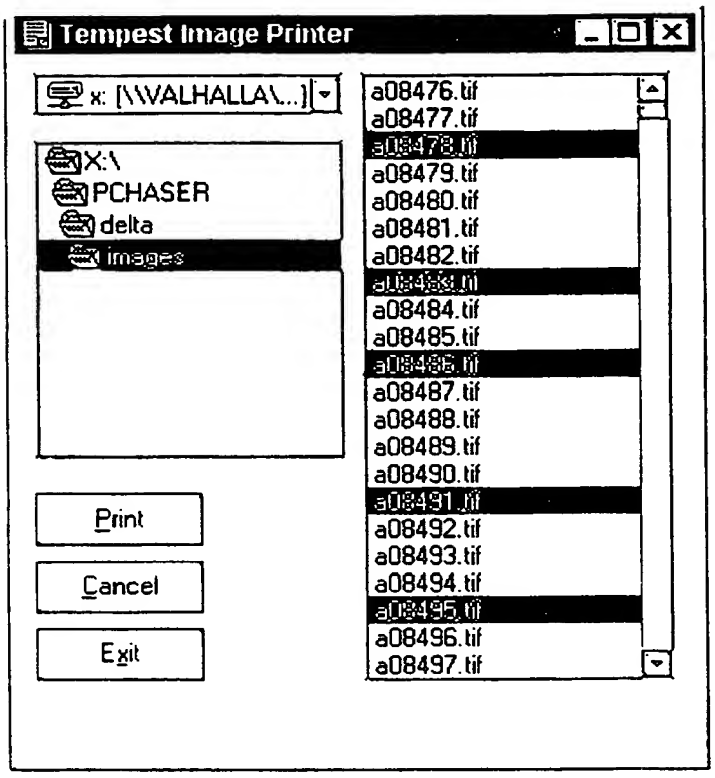


FIG 20

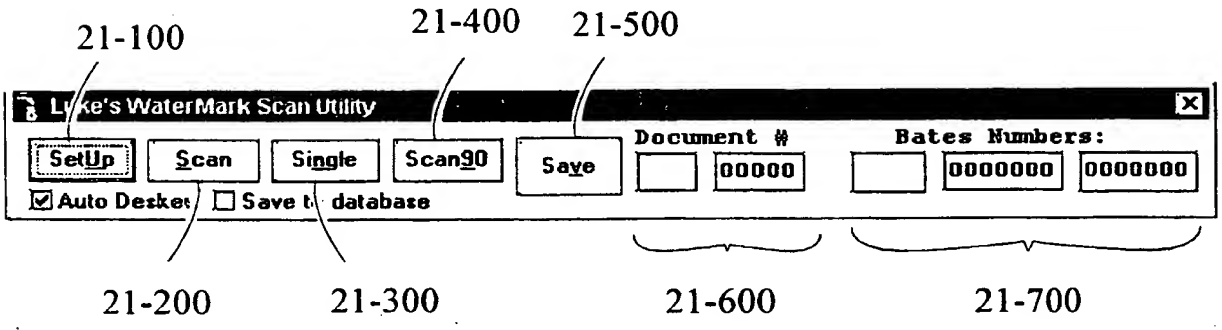


FIG 21

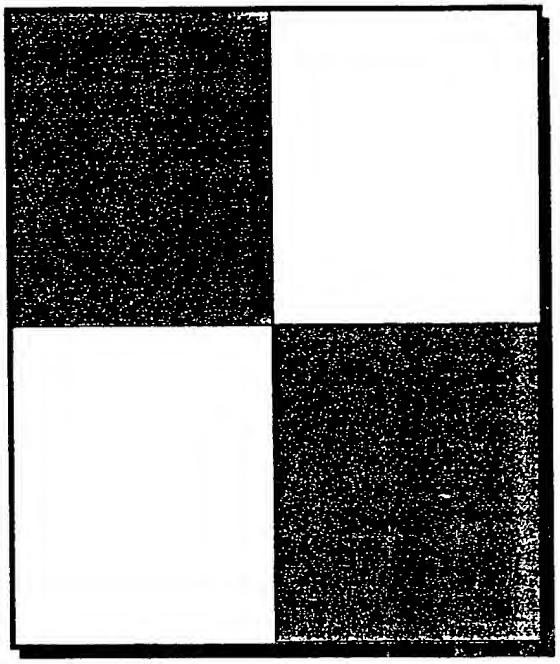


FIG 22

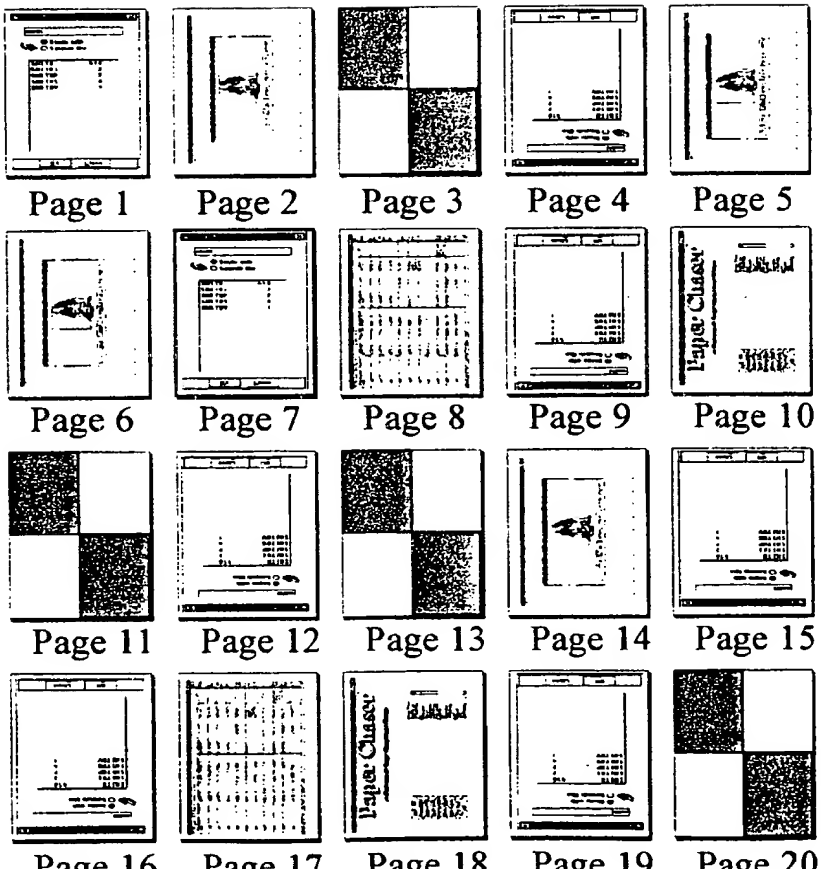


FIG 23

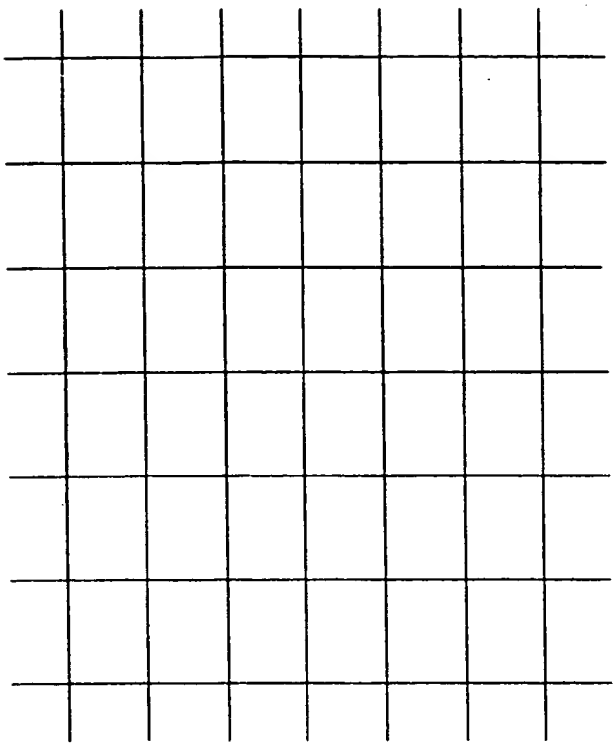


FIG 24

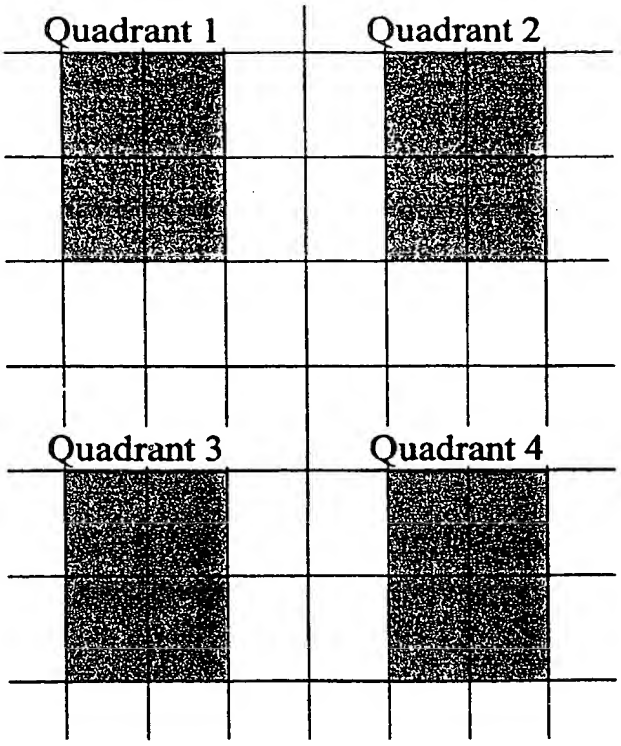
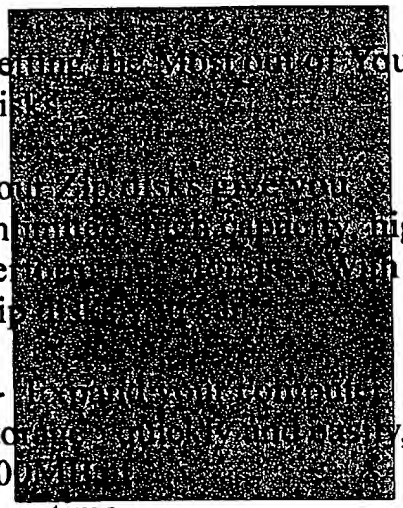


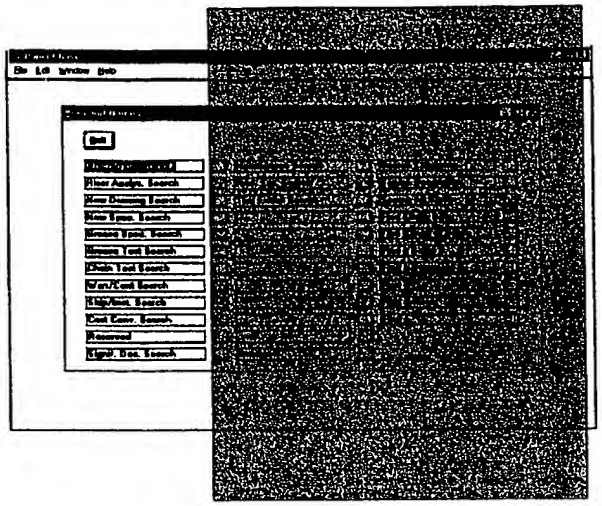
FIG 25



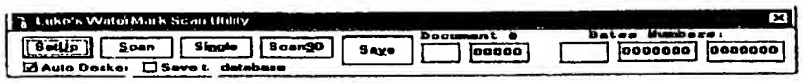
Ge[illegible]ur Zip Di[illegible]

Yo[illegible] un[illegible] pe[illegible] your Zi[illegible]

- [illegible] st[illegible] 10[illegible] a time.



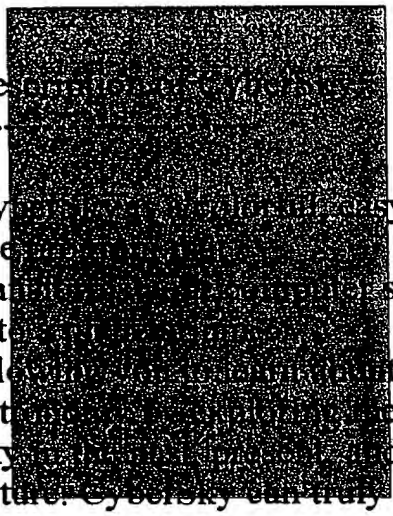
- Move your work easily to different locations and computers.



- Back up your hard disk or any

README.TXT

CyberSky Shareware Version 1.0d
Thursday, August 24, 1995



De[illegible]

Cy[illegible]y-to-use [illegible] tra[illegible]screen int[illegible] all[illegible] ast[illegible] sky[illegible] futu[illegible] be thought of as a desktop planetarium, because it

2246
gMoleculeSynthesized 1
gMoleculeAnalyzed 1
gVolume 5
gStarttimeInMaze 0
gUsingOxygenMask 1
gNoradLaun[illegible]
gUseRetinal[illegible]
gHistoricalL[illegible]
gAresMemo[illegible]
gMercuryM[illegible]
gPoseidonM[illegible]
gCalledFlFo[illegible]
gCommand[illegible]
gInitialRip 1
gPodAtUppe[illegible]
gReadyRoom[illegible]
gRobotDow[illegible]
gTSAElevatorBlink 1
gBkgndMonScore 500

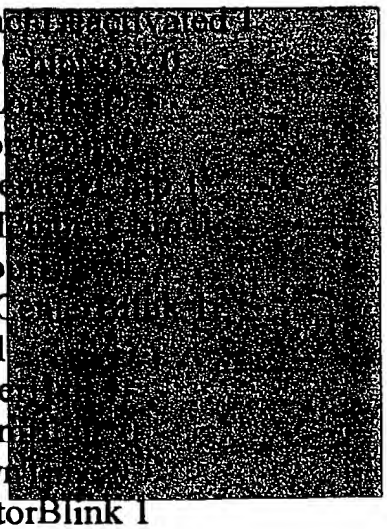


FIG 26

SUBSTITUTE SHEET (RULE 26)

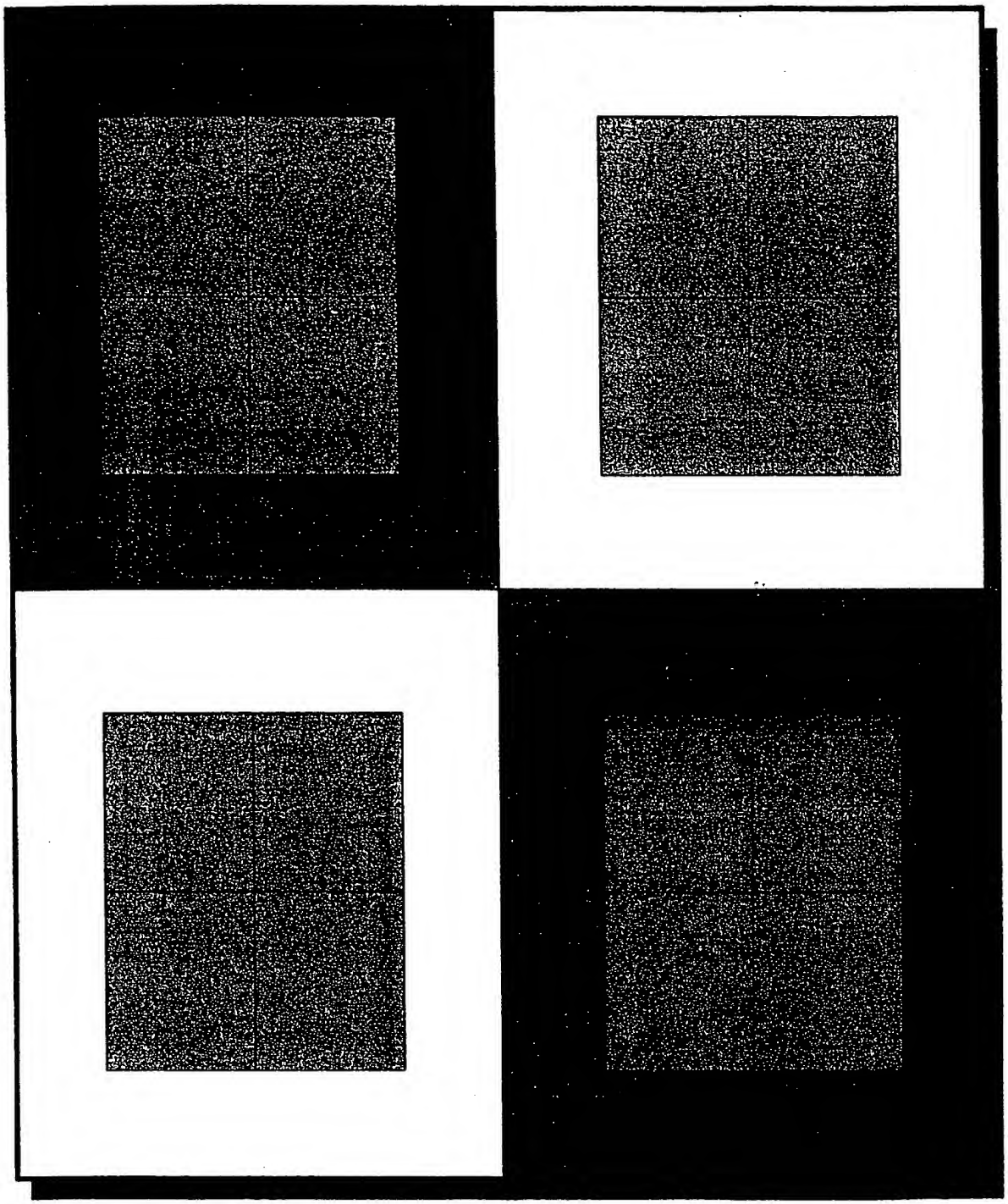
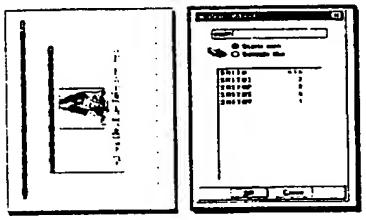
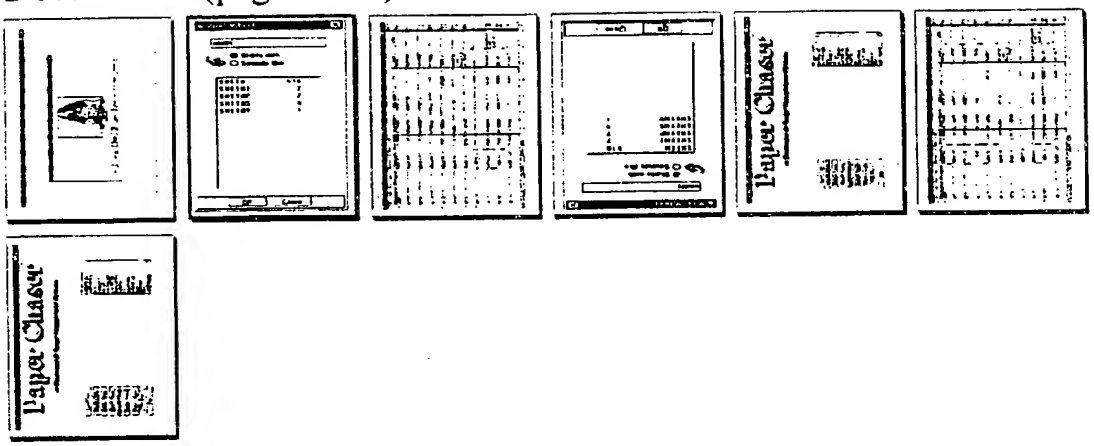


FIG 27

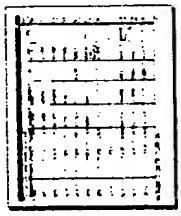
Document 1 (pages 1 & 2)



Document 2 (pages 4-10)



Document 3 (page 12)



Document 4 (pages 14-19)

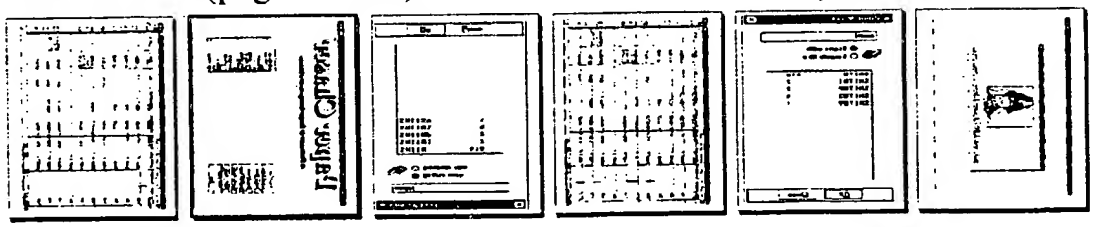


FIG 28

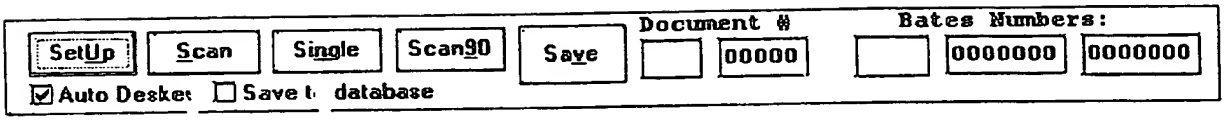


FIG 29

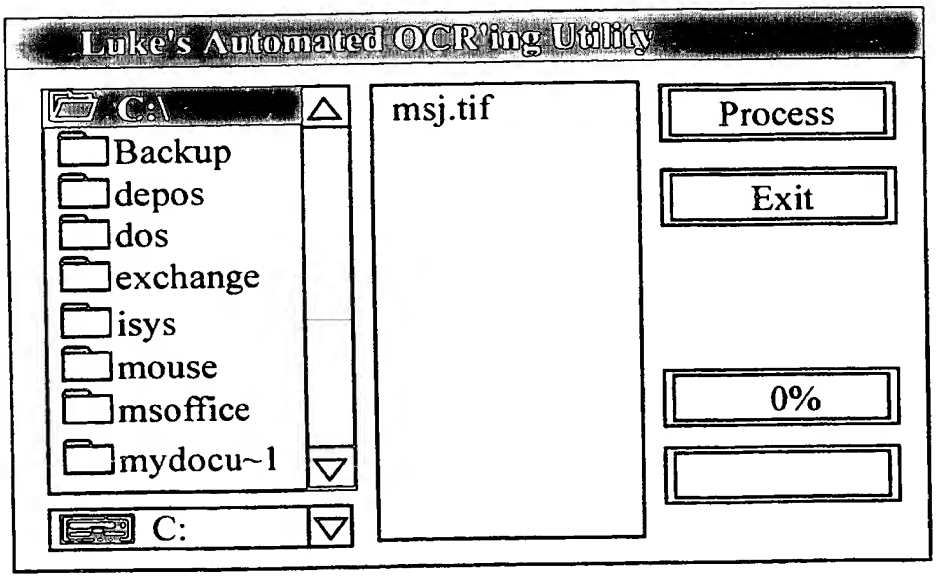


FIG 30

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 17/30

A3

(11) International Publication Number: WO 98/18092

(43) International Publication Date: 30 April 1998 (30.04.98)

(21) International Application Number: PCT/US97/18935

(22) International Filing Date: 21 October 1997 (21.10.97)

(30) Priority Data:

60/029,425	22 October 1996 (22.10.96)	US
60/028,985	22 October 1996 (22.10.96)	US

(71) Applicant *(for all designated States except US)*: TEMPEST SOFTWARE INCORPORATED [US/US]; Texas Commerce Tower, 50th floor, 600 Travis, Houston, TX 77002 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: SALTSMAN, Michael, L. [US/US]; 7306 Wovenwood Drive, Houston, TX 77041 (US). SPENCE, Luke, A. [US/US]; Apartment 2901, 1617 Fannin, Houston, TX 77002 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published

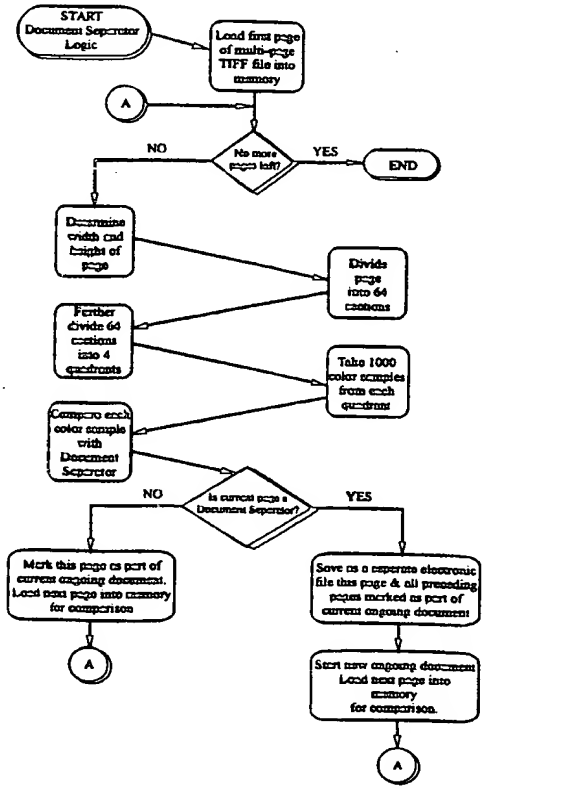
With international search report.

(88) Date of publication of the international search report: 20 August 1998 (20.08.98)

(54) Title: METHOD AND APPARATUS FOR SCANNING AND MANAGING DOCUMENT IMAGES

(57) Abstract

A document management system wherein efficient and high-speed inputting of a voluminous number of documents is facilitated by means of a document separator, where said document separator contains a predetermined unique graphic image, which said image is interpreted by said system to perform a predetermined set of tasks. A document management system wherein said system is modular in design and permits user to select individual software components to be used with said system.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

INTERNATIONAL SEARCH REPORT

rnational Application No
PCT/US 97/18935

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F17/30

According to International Patent Classification(IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category '	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 435 314 A (TOKYO SHIBAURA ELECTRIC CO ;TOSHIBA INTELLIGENT TECH (JP)) 3 July 1991 see the whole document ---	1-12,18
X	US 5 129 016 A (MURAKAMI TATSUYA ET AL) 7 July 1992 see the whole document ---	1-12,18
X	GA COTE R ET AL: "PROFILES IN DOCUMENT MANAGING" BYTE, vol. 17, no. 9, 1 September 1992, pages 198-200, 202, 204, 206 - 208, 210 - 212, XP000294465 see the whole document ---	13,15,16
Y	-/--	14

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

' Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
8 May 1998	10.06.98

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Abbing, R

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

ernational Application No
PCT/US 97/18935

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP 0 592 079 A (SUN MICROSYSTEMS INC) 13 April 1994 see abstract ---	17
Y	"AUTOMATIC TILING OF ARBITRARY OBJECTS" RESEARCH DISCLOSURE, no. 344, 1 December 1992, page 943 XP000327169 see the whole document -----	14

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 97/ 18935

Box I Observations where certain claims were found unsearchable (Continuation of item 1 of first sheet)

This International Search Report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the International Application that do not comply with the prescribed requirements to such an extent that no meaningful International Search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box II Observations where unity of invention is lacking (Continuation of item 2 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

CLAIMS: 1-12, 18
CLAIMS: 13-17

1. ☒ As all required additional search fees were timely paid by the applicant, this International Search Report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this International Search Report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☐ No required additional search fees were timely paid by the applicant. Consequently, this International Search Report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest.

☒ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT

Information on patent family members

ternational Application No
PCT/US 97/18935

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0435314 A	03-07-91	JP 3202966 A	04-09-91
		US 5081688 A	14-01-92
US 5129016 A	07-07-92	JP 2112797 C	21-11-96
		JP 8027831 B	21-03-96
		JP 62267876 A	20-11-87
		KR 9507393 B	10-07-95
EP 0592079 A	13-04-94	JP 6222910 A	12-08-94
		US 5555416 A	10-09-96